НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

«На правах рукопису» «До захисту допущено»

УДК 004.8

Завідувач кафедри
_____ *Стіренко С.Г.*
(підпис) (ініціали, прізвище)

“____” _____ 20***20*** р.

Магістерська дисертація

зі спеціальності: *123. Комп’ютерна інженерія*
(код та назва напряму підготовки або спеціальності)

Спеціалізація: *123. Комп’ютерні системи та мережі*

на тему: Програмний комплекс ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури

Виконав: студент VI курсу, групи ІВ-91мп
(шифр групи)

Юрченко Назар Олегович
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник проф., д.т.н., проф Писарчук О.О..
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант проф., д.т.н., проф.Кулаков Ю.О.
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет (інститут) *Інформатики та обчислювальної техніки*
(повна назва)

Кафедра *Обчислювальної техніки*
(повна назва)

Рівень вищої освіти – другий (магістерський) за освітньо-професійною програмою

Спеціальність *123. Комп'ютерна інженерія*
(код і назва)

Спеціалізація *123. Комп'ютерні системи та мережі*
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
Стіренко С.Г.
(підпис) (ініціали, прізвище)

« » 20***20*** р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Юрченку Назару Олеговичу
(прізвище, ім'я, по батькові)

1. Тема дисертації Програмний комплекс ідентифікації нештатних ситуацій на об'єктах критичної інфраструктури

Науковий керівник дисертації проф., д.т.н., Писарчук Олексій Олександрович
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «26» жовтня *2020* р. № 3132-с

2. Строк подання студентом дисертації 26.11.2020

3. Об'єкт дослідження процес автоматизованого виявлення та ідентифікації нештатних ситуацій на об'єктах критичної інфраструктури

4. Предмет дослідження методи теорії розпізнавання образів

5. Перелік завдань, які потрібно розробити: Аналіз відомих методологічних основ, аналіз відомих технологічних рішень, розробка методики та

технології, розробка програмного комплексу, оформлення результатів розробки програмного комплексу у формі стартап-проекту.____

6. Консультанти розділів дисертації:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	проф., д.т.н., проф.Кулаков Ю.О.		

7. Дата видачі завдання 18.12.2019

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів дисертації	Примітка
1.	*Затвердження теми роботи*	*18.12.2019*	
2.	*Вивчення та аналіз завдання*	*20.09.2020*	
3.	*Розробка архітектури та загальної структури систем*	*30.09.2020*	
4.	*Розробка структур окремих підсистем*	*2.10.2020*	
5.	*Програмна реалізація системи*	*12.10.2020*	
6.	*Оформлення пояснювальної записки*	*18.10.2020*	
7.	*Передзахист*	*26.11.2020*	
8.	*Захист*	*18.12.2020*	

Студент _____ _____
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ _____
(підпис) (ініціали, прізвище)

РЕФЕРАТ

на магістерську дисертацію

виконану на тему: Програмний комплекс ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури

студентом: Юрченко Назаром Олеговичем

Робота складається із вступу та чотирьох розділів. Загальний обсяг роботи: 90 аркушів основного тексту, 35 ілюстрації, 26 таблиць. При підготовці використовувалася література з 52 різних джерел.

Актуальність. Громадська безпека є головною проблемою сучасного суспільства. Сучасна зброя та вогнепальна зброя становлять серйозну загрозу безпеці повсякденного життя, а останні події та висвітлення у ЗМІ лише додатково оприлюднили властиві небезпеки, з якими можна зіткнутися навіть у найбільш публічних місцях. Тому виникає необхідність вирішення цієї проблеми, наприклад методом встановлення відеокамер в громадських місцях. Камери відеоспостереження також мають певні алгоритми обробки даних, за рахунок яких виявляються нештатні ситуації.

Ідея даної технології не є новою, проте активне впровадження технології припадає лише на останні 8 років. Такий метод пошуку злочинців є досить надійним, адже він може, як виявляти автоматично правопорушення, так і допомагати живій людині виконувати цю складну роботу.

Мета і завдання дослідження. Метою магістерської роботи є підвищення оперативності виявлення та ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури.

Для досягнення мети дослідження поставлено і вирішено такі завдання:

- Аналіз існуючих систем управління транзакціями в мережі Blockchain;
- Аналіз відомих технологічних рішень щодо оперативного виявлення нештатних ситуацій на об’єктах критичної інфраструктури;
- Розробка методики та технології виявлення нештатних ситуацій на об’єктах критичної інфраструктури;
- Розробка програмного комплексу ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури;
- Оформлення результатів розробки програмного комплексу ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури у формі стартап-проекту.

Об’єкт дослідження – процес автоматизованого виявлення та ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури.

Предмет дослідження – методи теорії розпізнавання образів.

Наукова новизна одержаних результатів роботи полягає у наступному:

Запропоновано спосіб ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури, який буде достатньо ефективно розпізнавати холодну та вогнепальну зброю, а також виконувати цю дію достатньо швидко

Особистий внесок здобувача. Магістерське дослідження є самостійно виконаною роботою, в якій відображено особистий авторський підхід та особисто отримані теоретичні та практичні результати, тобто було розроблено систему, що розпізнає холодну та вогнепальну зброю швидше та ефективніше ніж існуючі. Практичні

навички здобуті в дослідженні систем допомогли реалізувати даний програмний продукт.

Практична цінність. Даний продукт можна інтегрувати в різних місцях, де необхідна ідентифікація зловмисників, що намагаються зробити яке-небудь правопорушення з використанням холодної або вогнепальної зброї.

Публікації:

Юрченко Н.О. БЕЗПЕКА ГРОМАДСЬКИХ МІСЦЬ ЗА ДОПОМОГОЮ НЕЙРОМЕРЕЖ. *Збірник наукових матеріалів LV Міжнародної інтернет* —конференції «ПЕРСПЕКТИВНІ НАПРЯМКИ НАУКОВОГО ДОСЛІДЖЕННЯ» 24 листопада 2020 р., м.Львів http://el-conf.com.ua/

Ключові слова:

Нейронні мережі, Комп’ютерний зір, YOLOv3, Громадська безпека.

Abstract

For a master's thesis

Performed on the topic : Software solution for identifying emergency situations at critical infrastructure facilities

By student: Nazar Yurchenko

The work consists of an introduction and four sections. Total volume of work: 90 sheets of the main text, 35 illustrations, 26 tables. Literature from 52 different sources was used in the preparation.

Topicality. Public safety is a major problem in modern society. Modern weapons and firearms pose a serious threat to the security of everyday life, and recent events and media coverage have only further revealed the inherent dangers that can be encountered even in the most public places. Therefore, there is a need to solve this problem, for example, by installing video cameras in public places. Surveillance cameras also have certain data processing algorithms that detect abnormal situations.

The idea of this technology is not new, but the active implementation of the technology occurs only in the last 8 years. This method of finding criminals is quite reliable, because it can both automatically detect offenses and help a living person to perform this complex job.

The purpose and objectives of the study. The purpose of the master's work is to increase the efficiency of detection and identification of abnormal situations at critical infrastructure.

To achieve the goal of the study, the following tasks were set and solved:

- Analysis of existing transaction management systems in the Blockchain network;
- Analysis of known technological solutions for the rapid detection of abnormal situations at critical infrastructure;

- Development of methods and technology for detecting abnormal situations at critical infrastructure facilities;
- Development of a software package for the identification of abnormal situations at critical infrastructure facilities;
- Registration of results of development of the software complex of identification of emergencies on objects of critical infrastructure in the form of the startup project.

Object of study – the process of automated detection and identification of abnormal situations at critical infrastructure facilities

Subject of study – methods of theory of pattern recognition.

The scientific novelty of the obtained results is as follows:

A method of identifying abnormal situations at critical infrastructure facilities, which will be sufficiently effective to recognize cold steel and firearms, as well as perform this action quickly enough.

Personal contribution of the applicant. The master's research is a self-performed work that reflects the personal author's approach and personally obtained theoretical and practical results, i.e. a system has been developed that recognizes cold steel and firearms faster and more efficiently than existing ones. Practical skills gained in the study of systems helped to implement this software product.

Practical value. This product can be integrated in various places where it is necessary to identify criminals trying to commit any offense using firearm.

Publications:

Yurchenko N.O. SECURITY OF PUBLIC PLACES BY USING NEURAL NETWORKS. Collection of scientific materials of the LV International Internet Conference "PERSPECTIVE DIRECTIONS OF SCIENTIFIC RESEARCH" November 24, 2020, Lviv http://el-conf.com.ua/

Keywords:

Neural networks, Computer vision, YOLOv3.

ЗМІСТ

ВСТУП

На сьогоднішній день кількість правопорушень зростає з кількістю населення. Тому одним з рішень цього є встановлення камер в громадських місцях, що відстежують правопорушення, як приклад збройні пограбування. Здебільшого кадри відеоспостереження використовуються як докази вчинення злочину. Також камери використовуються в реальному часі. Але не зважаючи на те, що за зловмисником спостерігають камери відеоспостереження, оператор може не звернути увагу через наявність людського фактору. Ефективність аналізу відеопотоку може бути покращена шляхом обробки зображень штучним інтелектом та включення виявлення об'єктів у процес моніторингу.

Цінність кадрів відеоспостереження як доказу залежить від візуальної якості використовуваних камер. Тому спостерігається різке збільшення встановлення якісних камер відеоспостереження на громадських площах. Лише у Лондоні на сьогоднішній день для спостереження використовують близько 691-го мільйону камер відеоспостереження. На кінець 2010 року у Швеції було встановлено понад 50 000 камер. Ці камери відеоспостереження встановлюються в різних громадських місцях, таких як магазини, громадський транспорт, таксі, школи та фінансові установи. У Познані (Польща) за допомогою встановлення 450 камер уряд досяг успіху у зменшенні вуличних бійок на 40%, а випадків наркотиків - на 60%.

Алгоритми виявлення об’єктів, раніше впроваджені в відеоаналіз відеоспостереження, виявляють пішоходів, тварин та транспортні засоби. Ці алгоритми можуть бути розширені для виявлення людини, що тримає зброю, як вогнепальну зброю, або гострі предмети, такі як ножі, в громадських місцях або місцях обмеженого доступу.

Оскільки зброя в руках людини вважається більшою загрозою порівняно зі зброєю самою по собі, то виявлення людини на зображенні до виявлення зброї насправді являється вигідним. Є декілька відомих методів виявлення зброї.

Наприклад можна виконувати " віднімання фону" для виявлення людини до виявлення зброї, а потім вже виявлення людей.

На сьогоднішній день найефективнішим алгоритм розпізнавання об’єктів являється алгоритм YOLO, що розшифровується You Only Look Once, цей алгоритм ефективний тим, що може працювати зі швидкістю близько 30 кадрів в секунду. Цей алгоритм не зчитує кожну частину зображення, але спочатку визначає області, які треба переглянути. Для цього визначається сітка 13 на 13 і запускається алгоритм для ідентифікації областей з найвищою достовірністю, визначеною в конфігурації. Як тільки він ідентифікує надзвичайно впевнену область, він буде переданий наступним шарам для процесу виявлення об’єкта. Це змусило модель обробляти лише один раз, отже вона стала дуже швидкою.

Таким чином популяризація та запровадження цього відеоаналізу буде мати великий результат. Як приклад, камери будуть запобігати вуличним бійкам, а відповідно дадуть змогу реагувати службам охорони на злочинників завчасно.

РОЗДІЛ 1
АНАЛІЗ ВІДОМИХ МЕТОДОЛОГІЙ ТА ТЕХНОЛОГІЙ ІДЕНТИФІКАЦІЇ НЕШТАТНИХ СИТУАЦІЙ НА ОБ'ЄКТАХ КРИТИЧНОЇ ІНФРАСТРУКТУРИ.

На сьогоднішній день зростає кількість населення нашої планети, а саме зростає кількість людей, що проживають на планеті. Це також означає що густина населення зростає також. Разом з ростом населення росте кількість недобросовісних громадян, які здатні з різних причин порушувати закон. При порушенні закону, часто страждають ті чи інші люди, тому з'явилася потреба в методах для спостереження за населенням, а саме за своєчасне визначення злочинності. За рахунок цієї потреби з'явилися камери відеоспостереження, металодетектори, і подібні пристрої. Але так як зростає кількість населення, а відповідно його густина, то людей повз них проходить все більше і більше, що ускладнює роботу людини, що спостерігає за камерами.

1.1. Аналіз відомих методологічних рішень ідентифікації нештатних ситуацій на об'єктах критичної інфраструктури.

Існують різні методи знаходження зброї у зловмисника на об'єктах критичної інфраструктури. Наприклад на сьогоднішній день використовують такі речі як металодетектор, обслуговуючий персонал, який може проводити різного плану обшуки. Такі люди перевіряють на наявність заборонених речей у тих чи інших осіб на тих чи інших об'єктах. Також існують камери відеоспостереження, які подають на якогось співробітника доволі велику кількість зображень з різних точок місцевості, що перебуває під охороною. Таким співробітникам треба бути досить довгий час дуже уважними, щоб не упустити малих деталей, знання про які можуть допомогти запобігти того чи іншого виду злочин.

1.1.1. Рентгенівські промені на об’єктах критичної інфраструктури

Усім відомо, що в аеропортах стоять спеціальні сканери багажу, що сканують вміст різних сумок без їх відкривання. Такими системами обладнані не тільки аеропорти, а також деякі інші об’єкти критичної інфраструктури, але розглянемо безпосередньо випадок з аеропортами.

Метою перевірок безпеки в аеропорту є запобігання потраплянню небезпечних вантажів у повітряне судно та забезпечення безпечності всіх рейсів до місця призначення. Тому всі пасажири повинні пройти крізь металошукач перед тим, як увійти в зону очікування, а ручний багаж та зареєстрований багаж скануються за допомогою рентгенівського апарату. Про металошукач описано детальніше пізніше.

Дана технологія заснована на основі рентгенівських променів.

З точки зору пасажира, рентгенівський сканер виглядає як коробчастий тунель та конвеєрна стрічка, яка рухає багаж через тунель. Усередині коробки багаж сканується за допомогою рентгенівських променів: сканування стає можливим завдяки тому, що рентгенівські промені різною мірою проникають в різні речовини.

Машина виробляє рентгенівські промені за допомогою спеціальної трубки, яка покрита свинцем. У свинцевій підкладці є вузький зазор шириною близько одного сантиметра, через який рентгенівські промені спрямовуються в тунель. Стрічка транспортує кожен предмет багажу через рентгенівський промінь, а на протилежній стороні тунелю детектор вимірює кількість випромінювання, яке проникло через відсканований предмет. Щільні речовини, такі як свинець, поглинають найбільше випромінювання, блокуючи просування рентгенівських променів. Далі на підставі кількості випромінювання, яке пройшло багаж, комп’ютер формує зображення предметів, близьке до реальності.

Чим щільніше речовина або чим товщі шар речовини на шляху рентгенівського випромінювання, тим темніше предмет з’являється на екрані

комп’ютера. Речовини, занадто щільні, щоб радіація могла проходити крізь них, здаються чорними.

Раніше рентгенівські знімки були чорно-білими. Сьогодні різні матеріали відображаються різними кольорами на зображенні, що створено комп’ютером, прикріпленим до сканера. Органічні речовини, такі як дерево, вода, пластик та текстиль, мають оранжевий колір. Неорганічні речовини, такі як метали, мають блакитний колір. [1]

1.1.2. Металошукачі та металодетектори

Металошукачі використовують магнітні поля для ідентифікації металевих предметів. Магнітні поля створюються потоком струму через дроти або електричні прилади. Металошукачі створюють магнітне поле за допомогою короткого імпульсу електричного струму. Магнітне поле відбиватиметься на машині, якщо в ньому є металеві предмети, такі як годинник або пряжка ременя. Машина виявляє зворотний сигнал, і видається звуковий сигнал для попередження працівника. Металошукачі ігнорують дуже малу кількість металу, наприклад, ґудзик на джинсах або маленькі сережки. У деякому обладнанні використовується неіонізуюче випромінювання. Неіонізуюче випромінювання має достатньо енергії для переміщення атомів у молекулі навколо або змушує їх вібрувати, але недостатньо для видалення електронів з атомів. В аеропортах металошукачі та пристрої з міліметровими хвилями використовують низьку енергію, неіонізуюче випромінювання для передачі енергії через відсканові поверхні. Енергія, яка відскакує від поверхні, що сканується, покаже наявні об’єкти, або вона може генерувати зображення, яке робітники можуть використовувати для виявлення предметів, які можуть потребувати додаткових досліджень.

Пристрої міліметрових хвиль використовують неіонізуючі радіочастотні хвилі для виявлення металу. Пристрій випромінює хвилі, що відбиваються від тіла і повертаються до назад пристрою. Дані пристрої випромінюють набагато

менше енергії, ніж може випромінювати мобільний телефон. Пристрої міліметрових хвиль є важливим елементом обладнання безпеки аеропорту, оскільки вони можуть демонструвати приховані загрози, такі як пістолети та ножі. Якщо при скануванні когось, хто не має зброї чи інших потенційних загроз, екран стає зеленим і відображається "OK" (рис. 1.1). Якщо об’єкт виявлено, він з’явиться на екрані разом із загальним контуром тіла, щоб показати місцезнаходження. Не все, що відображається при скануванні, є фактичним об'єктом або загрозою, тому працівники виконують перевірку після сканування.[2]

Рис 1.1. Відображення результату сканування за допомогою сканеру [2]

1.1.4. Камери відеоспостереження

На сьогоднішній день у громадських місцях встановлені камери відеоспостереження. Такі камери використовують для різних цілей, зокрема для відстеження незаконних дій, таких як збройні пограбування. Здебільшого кадри з камер відеоспостереження використовуються як докази вже після здійснення злочину. Також у багатьох випадках існує спеціальний персонал, що займається спостереженням за місцями з камер відеоспостереження, але при цьому даний персонал може відволікатися, і складно постійно бути повністю в увазі до камер. Ефективність відеоспостереження може бути підвищена шляхом вбудовування алгоритмів обробки зображень та виявлення об’єктів у процес моніторингу.

Алгоритми виявлення об’єктів, раніше впроваджені у відео-аналіз відеоспостереження, вже можуть виявляти пішоходів, тварин та транспортні засоби. Ці алгоритми можуть бути розширені певним чином. Наприклад виявляти не просто людей, а людей, що тримають вогнепальну зброю, або гострі предмети такі як ножі, в громадських місцях або на об’єктах критичної інфраструктури.

Камери відеоспостереження являються одним з найліпших засобів, що відповідають експлуатаційним вимогами, що слід враховувати в широких аспектах безпеки [3]. Камери відеоспостереження встановлюються майже скрізь у громадських місцях для забезпечення безпеки [4]. Основне використання камер відеоспостереження полягає у забезпеченні безпеки, стримування (забезпечення певних заходів, таких як сигналізація про наближення, системи виявлення зловмисників, тощо), розслідування злочинів та зменшення страхових витрат [5]. Методи камер відеоспостереження для стримування різняться для різних категорій злочинів [3].

Зображення з камер відеоспостереження вважаються одними з найважливіших доказів у правоохоронних органах та судах [5]. Цінність таких відеозаписів, які взяті з камер відеоспостереження, як доказів сильно залежить від візуальної якості використовуваних камер. Через це спостерігається різке збільшення встановлення більш якісних камер відеоспостереження на громадських площах [6] (рис 1.2). Лише у Лондоні на сьогоднішній день для спостереження використовують близько 691-го мільйону камер відеоспостереження [7]. На кінець 2010 року у Швеції було встановлено понад 50 000 камер [6]. Ці камери відеоспостереження встановлюються в різних громадських місцях, таких як магазини, системи громадського транспорту, таксі, школи та фінансові установи [6]. Це змусило громадськість почуватися в безпеці та зменшило страх перед злочинами на вулицях [3]. У Познані (Польща) за допомогою встановлення 450 камер уряд досяг успіху у зменшенні вуличних бійок на 40%, а випадків наркотиків – на 60% [4].

Рис. 1.2. Камери відеосопостереження часто ставлять на вулицях та навіть в жилих будинках.

В результаті збільшення кількості камер відеоспостереження, що використовуються для спостереження, кількість екранів, які слід контролювати оператором відеоспостереження, значно зросла. Як уже згадувалося раніше, одним із найважливіших застосувань камер відеоспостереження є стримування злочинності. Моніторинг декількох різних екранів або зображень на одному екрані одночасно є напруженим завданням для оператора камер відеоспостереження. Кількість зображень, які може контролювати кожен оператор, буде сильно залежати від різних людських факторів [4]. У міру збільшення кількості зображень, які слід контролювати кожним оператором, концентрація спостережень людини на кожному екрані одночасно різко зменшується з часом [8]. Для одного оператора досить складно спостерігати за декількома екранами одночасно протягом тривалого часу. Згідно з дослідженнями [9], концентрація оператора відеоспостереження коливається в якості 83%, 84% та 64% після години безперервного моніторингу 4, 9 та 16 екранів відповідно.

Тому зростає попит на пошук автоматизованих алгоритмів спостереження. Основна мета автоматизованого спостереження – попередити оператора відеоспостереження, коли виникає небезпечна ситуація, але при цьому не

відволікати на непотрібні моменти. Під небезпечною ситуацією розуміється особа або група осіб, які нападають, створюючи страх або хвилювання в громадських місцях небезпечною зброєю, такою як ножі чи пістолети. Концепція автоматизованого спостереження можлива за допомогою включення в камери алгоритмів виявлення об'єктів із області комп'ютерного зору та обробки зображень. Використання алгоритмів виявлення об'єктів при програмній обробці відеоматеріалів у камерах спостереження було розпочато в останні роки і, як правило, використовується в інтелектуальній транспортній системі для моніторингу дорожнього руху [10].

1.2. . Аналіз відомих технологічних рішень ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури

Тож в даній роботі вирішено порівняти декілька систем, за допомогою яких можна ідентифікувати нештатні ситуації на об’єктах критичної інфраструктури. Було прийнято рішення розглянути декілька ідей, які можна використати для точного та швидкого аналізу зображень відео-потоку.

Будуть розроблені та розглянуті наступні системи:

1. Алгоритми, що не використовують глибинне навчання;
2. Алгоритми глибинного навчання.

Тож маємо три методики, кожна з яких буде доповнюватися різними елементами. Для порівняльних характеристик вирішено було взяти існуючу нейронну мережу, і запустити її на апаратному забезпеченні на якому буде проходити порівняння.

1.2.1. *Active Appearance Model*

Дана система розроблена за принципом Active Appearance Model. Цей тип алгоритм, що використовується у комп’ютерному зорі для того, щоб визначити який-небудь об’єкт. Алгоритм працює за статистикою, тобто існують статистичні дані про форму та зовнішній вигляд об’єкту, і ці дані порівнюються

з новим зображенням для того, щоб виявити, чи містить нове зображення даний об’єкт, чи ні. Нейронні мережі, що використовують цей алгоритм для виявлення тих чи інших об’єктів, будуються на етапі навчання моделі. На зображеннях, які подаються на навчання визначають певні деталі, що притаманні тому чи іншому об’єктові, наприклад точки вершин об’єкту, в комп’ютерному зорі це всього-то координати на зображенні. Дана процедура робиться вручну, тобто навчається людиною. Таким чином набір зображень разом із координатами орієнтирів, які відображаються на всіх зображеннях подаються на нейронні мережі.

На рисунку 2.1 зображено послідовність роботи алгоритму моделі для виявлення того чи іншого об’єкту.

Де x – це середнє фігури, $P_s = \{S_i\}$ – це матриця ортонормованого базисного вектору, S_i описує режим варіацій навчального набору, і b_s – це параметр об’єкту.

Недоліком використання даної моделі є те, що вона сильно залежить від контрасту зображення. Важко виявити фігури на шумному зображенні, тобто зображенні з великою кількістю зайвої інформації. Підхід, який використовується для виявлення ножів, добре працює там, де наконечник ножа добре видно.

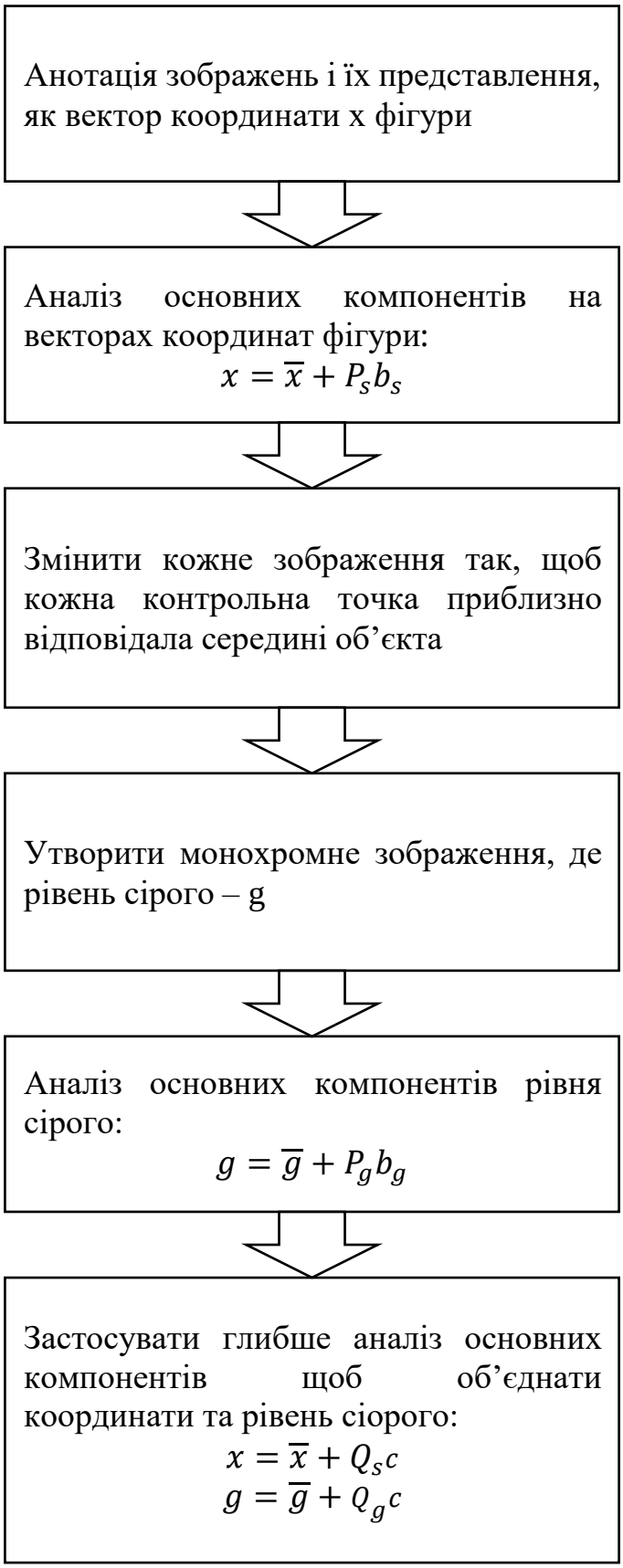


Рис. 2.1. Active appearance model - flow [34]

1.2.2. Детектор кутів Харріса

Алгоритм детектора кутів Харріса [35] дуже часто використовується для знаходження кутів на зображеннях. Вперше він був представлений в 1988 р. Етапи виявлення кутів є

1. Перетворення кольорового зображення у відтінки сірого
2. Обчислення просторової похідної
3. Налаштування тензорної структури
4. Розрахунок реакції Харріса
5. Зменшення Non-Max, щоб розглядати лише найближче.

Це алгоритм вилучення функції в одну точку. Саме різницю в інтенсивності у всіх напрямках для зміщення (u, v) можна показати у рівнянні 1.

$$f(\Delta x, \Delta y) = \sum_{(x_k, y_k) \in W} (I(x_k, y_k) - I(x_k + \Delta x, y_k + \Delta y))^2$$

Тут функцією вікна може бути гауссове вікно, яке надає ваги пікселям знизу, або вікно, яке є прямокутним. Для виявлення кутів *E(u,v)* слід максимізувати, і це можна зробити, застосувавши розширення Тейлора, яке дасть остаточне рівняння, зазначене як рівняння 2 і 3.

$$E(u, v) \approx [u \quad v] \, M \begin{bmatrix} u \\ v \end{bmatrix}$$

Де

$$M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

Тут I_x та I_y є похідними від зображення в напрямках (x, y), і нарешті ми вимірюємо кутову реакцію як рівняння 4.

$$R = \det(M) - k(trace\ (M))^2$$

Де

- det(M) = $\lambda_1\lambda_2$
- trace(M) = $\lambda_1 + \lambda_2$
- λ_1 та λ_2 являються власними значеннями *M*
- k – константа

І детектор кутів Харріса, і модель активного вигляду використовують рентгенівські знімки при виявленні ножів та пістолетів. Використання кутового детектора Харріса для пістолетів призводить до отримання більше хибних спрацьовувань з mAP 84,26% [36]. Поєднання обох алгоритмів виявлення ножів дає точність класифікації 92,5% [36].

1.2.3. Кольорова сегментація використовуючи кластеризацію методом к-середніх

Кластеризація методом к-середніх – це метод розподілу точок даних на кілька підмножин, також відомих як кластери [34]. Де кожна точка даних належить до підмножини з найближчим середнім значенням. Кількість кінцевих підмножин у вихідних даних представлено k. Функція втрат алгоритму кластерізації методом к-середніх є сумарною дисперсією всередині кластера, і вона визначається як рівняння 5.

$$E = \sum_{i=1}^{k} \sum_{x \in C_t} |\ x - \ m_i|^2$$

Де *x* – це будь-яка точка даних а m_i – середнє значення кластеру C_i.

Для виявлення пістолету виконується кольорова сегментація за допомогою алгоритму кластеризації методом к-середніх. Це усуне кольори, що не пов’язані між собою на зображенні. Як тільки непов’язані кольори видаляються, для виявлення об’єкта використовується детектор кутів Харріса. Цей метод досяг точності 84.26% [35].

У подальшій частині цього розділу буде описано основні алгоритми глибокого вивчення зображень, які використовуються для виявлення ножів та пістолетів.

1.2.4. Згорткова нейронна мережа

Точність виявлення зображень різко зросла з 2012 року після впровадження згорткових нейронних мереж (CNN) [37]. У згортковій нейронній мережі алгоритм зчитує зображення, створюючи сегменти, також відомі як матричні. Кожна матриця передається в кілька шарів. Ці шари включають максимальний рівень об’єднання та функцію активації. Функція шару максимального об’єднання полягає у вибіркових даних, лише враховуючи лише важливі особливості. Нарешті, після проходження цих шарів результат ідентифікується, а потім відповідає початковому класу. Для перевірки точності мережі використовуються функції помилок.

Функції помилок - це функція, яка використовується для обчислення відстані прогнозованих результатів від фактичного результату. Найбільш поширеними функціями помилок є середньоквадратична помилка, середня абсолютна помилка та логарифмічна втрата. Якщо рівень помилок занадто високий, він повернеться до першого шару та оновить ваги, де це потрібно. Щоб розрахувати, які ваги потрібно оновити, використовується техніка, відома як зворотне поширення. Процес повторюється доти, доки не буде встановлений відповідний рівень необхідної точності. Згорткові нейронні мережі досягли частоти помилок, встановлених тестом, 36,7% і 15,4% [37]. На Рисунку 2.2 зображена згорткова нейронна мережа з декількома шарами.

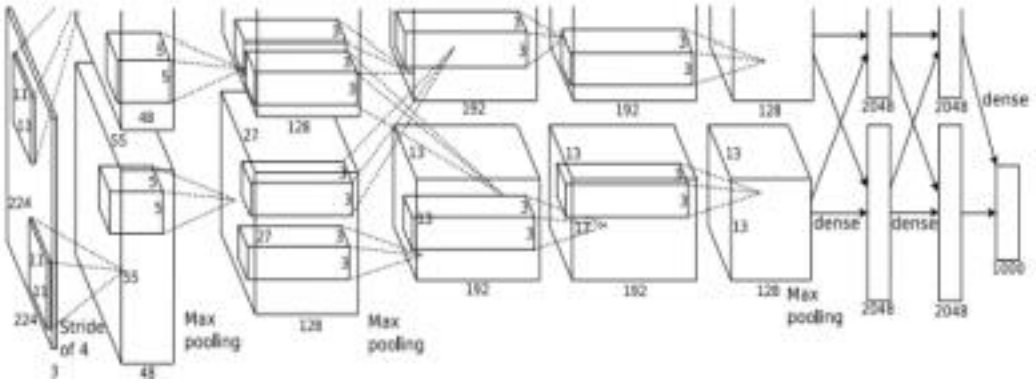


Рис 2.2. Згорткова нейронна мережа [34]

У роботі [36] дослідник навчав свою згорткову нейронну мережу, використовуючи зображення з дуже низькою роздільною здатністю для виявлення та розпізнавання людських обличь та зброї в режимі реального часу. Однак їх результати та обговорення більше зосереджуються на обличчі та не відображають жодної інформації щодо виявлення зброї [35]. Нещодавно автори використовували згорткову нейронну мережу для автоматичного виявлення холодної металевої зброї у яскравому відео спостереження [34].

Також в цій роботі запропонували метод зменшення складності та підвищення точності за допомогою використання інформації про краї, як ознаки для виявлення присутності вогнепальної зброї. Їхня техніка заснована на згортковій нейронній мережі для класифікації кадрів відеоспостережень, досягаючи виявлення точності 97,78%.

1.2.5. OverFeat

OverFeat - це згорткові нейронні мережі (CNN), що використовує підхід з зсувними вікнами [38]. У зсувному віконному детекторі, де класифікатор тренується спочатку на центральному зображенні, а потім класифікатор застосовується до кожного місця цільового зображення. Недоліком використання зсувних вікон є те, що він надзвичайно повільний, оскільки охоплює кожен піксель одного зображення і включає велику кількість (104)

вікон. Перевагою використання OverFeat є його точність. Через свою повільність його не можна використовувати в режимі реального часу виявлення об'єктів.

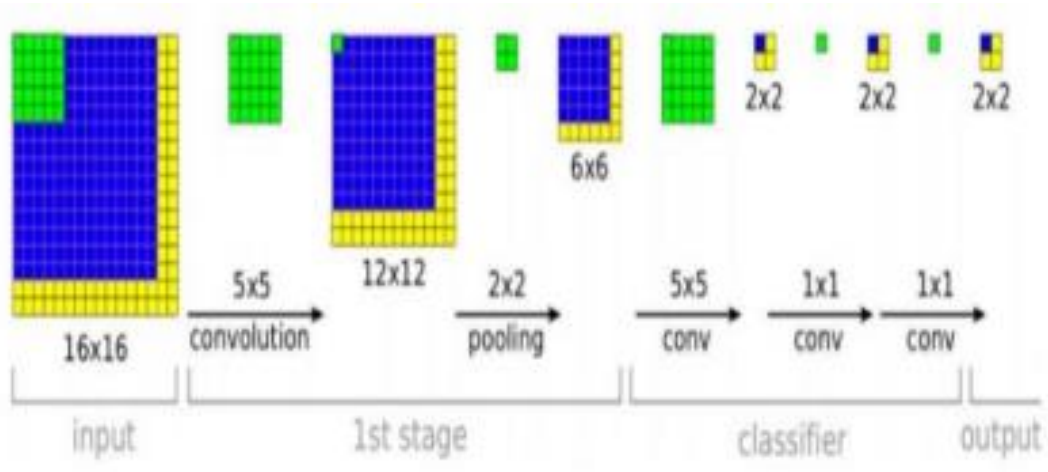


Рис 2.3. Зсувне вікно потребує додаткових обчислювальних можливостей, коли проходить повз усі шари [34]

1.2.6. R-CNN

Однією з головних проблем підходу до розсувного вікна є те, що він зчитує усі можливі області зображення. Предмет, що цікавить, може складатися з різних просторових розташувань та з різним співвідношенням сторін у межах зображення. Для цього буде потрібно величезна кількість регіонів для вибору та обробки. R-CNN обходить цю проблему і використовує вибірковий метод пошуку [36]. Цей метод виділяє 2000 регіонів, також відомих як регіональні пропозиції. Там області згинаються в квадрат і перенаправляються до згорткової нейронної мережі і генерують 4096-мірний векторний елемент. Ці функції передаються SVM для класифікації регіонів. Він також виконує алгоритм регресії для пошуку обмежувальних вікон класифікованого об'єкта, показаного на Рисунку 2.4.

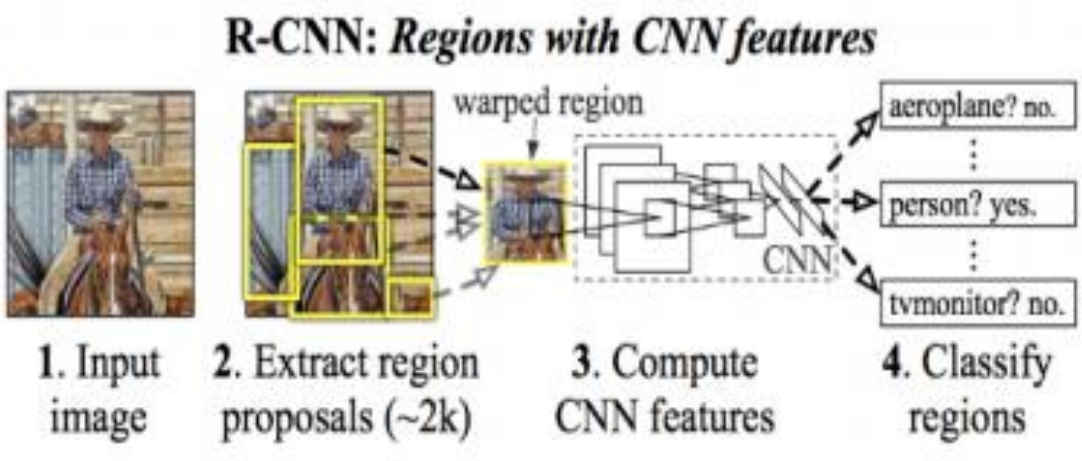


Рис.2.4. Послідовність виявлення об’єкту використовуючи R-CNN[34]

Проблеми цього підходу полягають у тому, що для навчання потрібен величезний час, оскільки він повинен класифікувати 2000 регіонів на зображення. Неможливо використовувати його в режимі реального часу, оскільки це займає приблизно 47 секунд на зображення.

1.2.7. Швидкі R-CNN

Швидкі R-CNN вирішили проблеми R-CNN та побудували на багато швидший алгоритм. Основний підхід у них абсолютно такий самий, як і до R-CNN, але в даному випадку цей алгоритм не вираховує область, але передає зображення безпосередньо до згорткової нейронної мережі та генерує карту особливостей. На цій згорнутій карті особливостей пропозиції регіону визначаються і деформуються на квадрати, а за допомогою шару об’єднання області інтересу, форма змінюється на фіксований розмір і подається на повністю з’єднаний шар. Шар softmax використовується для прогнозування класу та обмежувального поля з вектору функцій регіону інтересу, як показано на Рисунку 2.5. Оскільки ми не генеруємо 2000 запропонованих областей, цей алгоритм в рази швидший, ніж звичайний R-CNN.

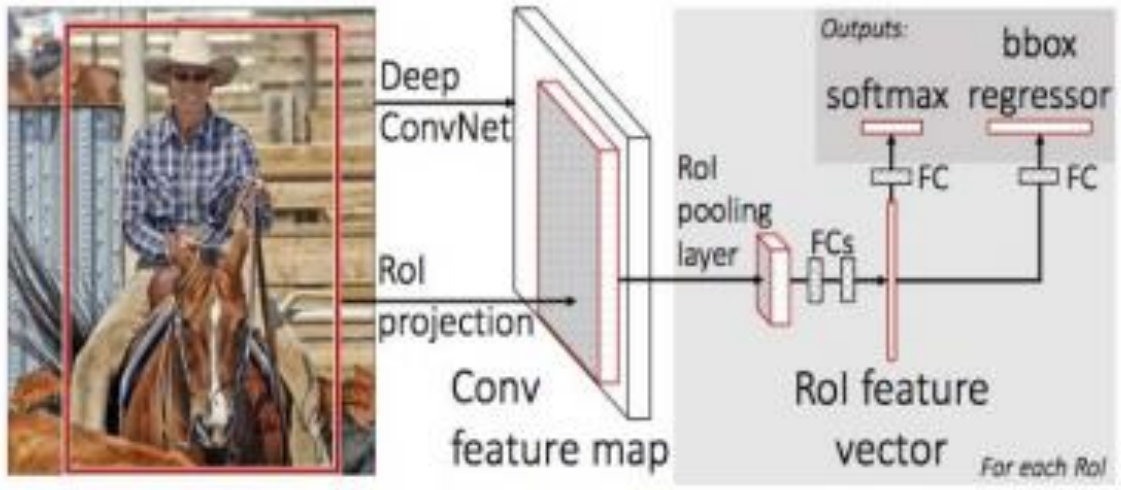


Рис.2.5. Архітектура швидкого R-CNN. [34]

1.2.8. Швидші R-CNN

Швидші R-CNN – це повністю згорткова нейронна мережа. Вона складається з двох нейронних мереж. Перша мережа генерує область пропозицій, а друга мережа використовує запропонований регіон, щоб виявляти та класифікувати об’єкти [34].

Наступна інформація отримується при передачі зображення на швидшу R-CNN:

1. Перелік обмежувальних зон
2. Маркер кожного обмежувального вікна
3. Імовірність кожного обмежувального вікна

Зображення завжди представлені у вигляді висоти, ширини та глибини та відомі як тензори. Потім тензор (зображення) проходить через згорткову модель, яка попередньо тренується, поки не досягне проміжного шару та не створить карту функцій, а потім переходить до мережі пропозицій областей. Мережа пропозицій областей (RPN) використовує функції, щоб запропонувати регіони, які можуть мати деякі об’єкти. Мережа пропозицій областей використовує маркери для вирішення проблеми змінної довжини. Вони являють собою області фіксованого розміру, розміщені на зображенні, що мають кілька відношень та розмірів, і будуть використовуватися для прогнозування розташування об’єктів.

Коли у нас є можливий перелік об’єктів з місцем розташування, стає дуже легко класифікувати об’єкти. Особливості CNN та областей об’єктів, що використовуються Регіоном об’єднань інтересів, та з’ясування об’єктів, близьких до об’єктів. Нарешті, модуль R-CNN використовує для класифікації об’єкта у вікні та коригування його координат.

На Рисунку 2.6. зображено послідовність роботи швидшої R-CNN з усіма шарами в мережі включаючи мережу пропозицій областей для вибору областей та класифікатор.

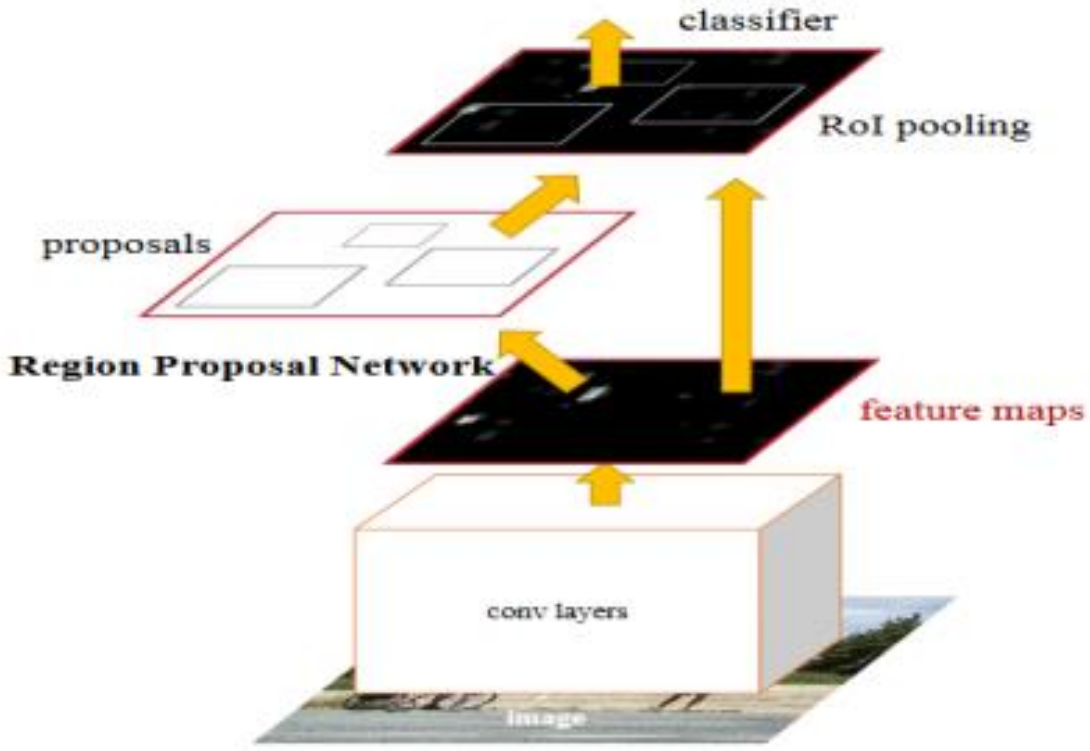


Рис 2.6.Архітектура швидшої R-CNN [34]

1.2.9. YOLO (You Only Look Once)

Більшість алгоритмів були розроблені після представлення згорткових нейронних мереж у 2012 році. Однак найновіший алгоритм під назвою YOLO що розшифровується You Only Look Once, досягнув продуктивності більше як 30 кадрів за секунду [39]. На відміну від CNN, він не зчитує кожну частину зображення, але спочатку визначає області, які треба переглянути. Для цього

малюють сітку 13 на 13 і запускають алгоритм для ідентифікації областей з найвищою достовірністю, визначеною в конфігурації. Як тільки він ідентифікує надзвичайно впевнену область, він буде переданий наступним шарам для процесу виявлення об’єкта. Це змусило модель обробляти лише один раз, отже вона стала дуже швидкою [39]. YOLO найбільш відомий своєю швидкістю, яка становить вище як 30 кадрів на секунду. До YOLO найшвидша була R-CNN, що мала найвищу швидкість в 7 кадрів за секунду [34].

На Рисунку 2.7 зображено як зроблено передбачення обмежувальної області. Де t_x, t_w, t_h – обмежувальні рамки. (c_x, c_y) – лівий кут зображення та p_w, p_h – ширина та висота зображення.

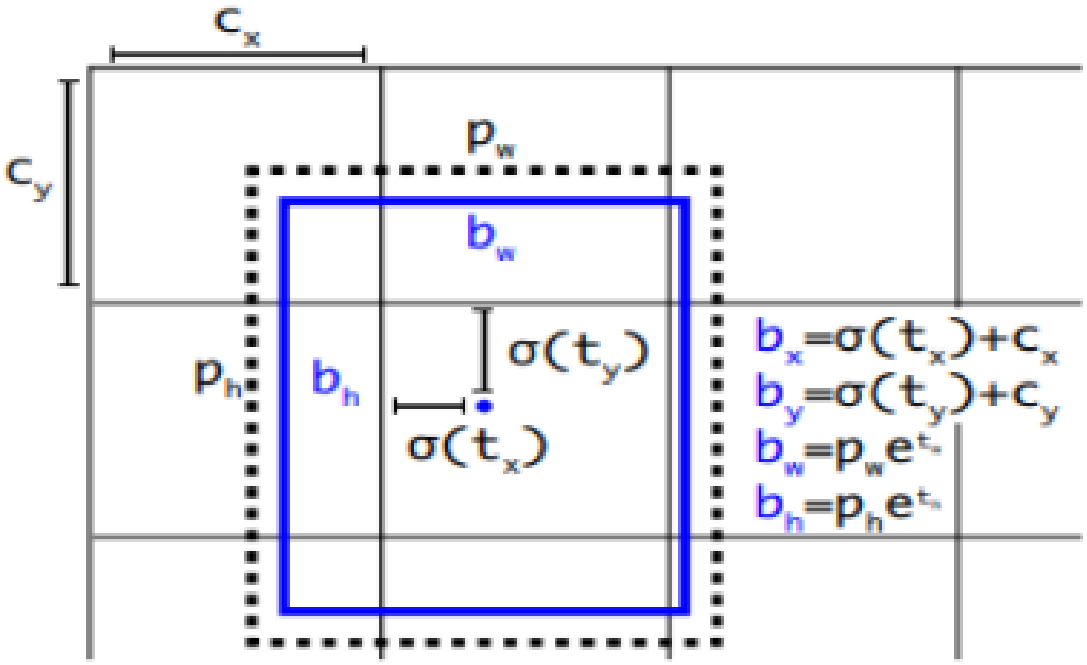


Рис. 2.7: обмежувальні області з пріоритетами розмірів та передбаченням розташування.

Середня точність (mAP) обчислюється з використанням кількості справжніх позитивних та справжніх негативних результатів, показаних у наступному рівнянні. Справжніми позитивними є ті, де алгоритм може успішно

виявити об'єкт. Помилково позитивним є кількість об’єктів, виявлених помилково.

$$\text{Точність} = \frac{\text{справжні позитивні}}{\text{справжні позитивні} + \text{справжні негативні}}$$

1.3. Критичні об’єкти системи спостереження та охорона критичних об’єктів

Спочатку слід розібратися, що таке об’єкти критичної інфраструктури. Раніше деякі з них вже згадувалися, але все ж таки, об’єктами критичної інфраструктури називають такі підприємства та установи наступних галузей:

1. Енергетика
2. Хімічна промисловість
3. Транспортна промисловість
4. Банки, фінанси
5. Охорона здоров’я
6. Продовольство

А також низка інших галузей. Згадані вище об’єкти являються стратегічно важливими в країні або державі для існування взаємодій між усіма сферами життєдіяльності. Від цього залежить безпека економіки, суспільства, населення та й навіть самої держави. Руйнування об’єктів критичної інфраструктури може вплинути на національну безпеку, охорону природного середовища, призвести до кризи ту чи іншу людину або сім’ю.[32]

Суспільні функції сильно залежать від мережевих систем у розвинених країнах світу. Навіть найосновніші повсякденні функції передбачають взаємодію з різноманітними критично важливими інфраструктурними системами. Наприклад, мільйони американців використовують транспортну інфраструктуру, щоб дістатися до роботи, школи чи місцевого торгового центру. Телекомунікаційна інфраструктура використовується для підтримки контакту з родиною та друзями, здійснення покупок або здійснення фінансових операцій.

Енергетична інфраструктура використовується для обігріву будинку, живлення місцевої промисловості та доставки палива до автомобілів. Хоча ці основні види діяльності відносно легко зрозуміти, масштаби використання інфраструктури менш очевидні. Наприклад, понад 19 мільярдів тон вантажів вартістю 13 трильйонів доларів США було переміщено через мультимодальну транспортну систему та пов'язані з нею мережі у США протягом 2002 року (USDOT, 2006). Що стосується телекомунікаційних мереж, то в 2002 році магістральний трафік США перевищував 100 петабайт на місяць (SVBJ, 2002). Якщо припустити, що середня електронна пошта становить 25 кілобайт, це перетворюється на 45 035 996 273 листів на місяць. Нарешті, добова потужність газової мережі США становить 5,6 мільярдів кубічних метрів [33]. Більше того, оскільки працездатність цих систем може бути вразливою до катастроф, аварій та навмисних збитків, існує потреба зрозуміти, наскільки критична інфраструктура та її функціональність можуть зазнати впливу під впливом порушень. Таким чином, існує потреба у розробці стратегій планування мережевих систем, здатних виживати та працювати під тиском

Висновки до розділу 1

У розділі 1 було розглянуто різні відомі методології та технології ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури. В результаті такого швидкого огляду, було зрозуміло, що на даний момент в світі існують досить таки хороші методи ідентифікації злочинності наприклад в аеропортах за допомогою металодетекторів, сканерів та камер відеоспостереження. Але якщо брати до уваги менш великі об’єкти критичної інфраструктури, то тяжко проводити повсякденний повний огляд кожної людини, особливо в досить таки наповнених місцях. Через це виникає необхідність розвивати рівень ідентифікації.

Було прийнято рішення, що найлегший метод суттєвого покращення ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури – це додати до обробки даних людини машину. А саме в тих точках, де людина може вправлятися досить добре, але за наявності людських факторів людина з плином часу справляється не дуже добре. Як можна було здогадатися, мова йде про відеоспостереження. Адже саме в цій сфері операторові доводиться дивитися постійно на мирні статичні картинки з різних камер, звикаючи, і погіршуючи увагу. Але саме в такі моменти може трапитися щось неочікуване і не добре.

Також відомо, що навіть машинне навчання не дозволить на всі 100% визначати нештатні ситуації принаймні через різні причини (зміна освітлення, краплі на камерах, тощо), то програмне забезпечення найліпше підходить саме для допомоги операторові розпізнавання тих чи інших ситуацій.

Тож було прийнято рішення розробити систему на основі комп’ютерного зору та глибинного навчання. Судячи з результатів попереднього дослідження це дозволить зробити систему достатньо точною, і імовірно достатньо швидкою. Також слід порівняти варіанти роботи різних систем на одній машині, адже ніяк інакше перевірити швидкодію та точність неможливо.

Так жоден варіант не був ідеальним в цій роботі. Це може бути через те, що архітектури нейронних мереж надто загальні та надто широко спрямовані. В

наступному розділі розроблюється система, головна суть якої спростити архітектуру нейронної мережі, задля прискорення її роботи. Очікується, що зброя на зображеннях являється зазвичай маленькою, тому немає потреби розглядати великі об’єкти, а лише малі.

РОЗДІЛ 2
РОЗРОБКА ПРОГРАМНОГО КОМПЛЕКСУ ІДЕНТИФІКАЦІЇ НЕШТАТНИХ СИТУАЦІЙ НА ОБ'ЄКТАХ КРИТИЧНОЇ ІНФРАСТРУКТУРИ

Як відомо, алгоритм структура нейронної мережі YOLO дуже добре підходить для виконання цієї задачі. Але порівнявши алгоритми YOLO другої версії та третьої версії, можна зробити висновок, що другої версії алгоритм досить таки швидкий, а третьої версії досить таки надійний. Так як задача ідентифікації нештатних ситуацій на об'єктах критичної інфраструктури в даному випадку полягає в знаходженні зброї у зловмисників в руках із камер відеоспостереження, то має сенс шукати саме дрібні об'єкти, а не намагатися знайти, об'єкт, що являється на більшу частину екрану.

Детальніше процес розробки та порівняння цих двох версій структури нейронних мереж YOLOv2, YOLOv3 та власної структури, що об'єднує найкращі характеристики обох цих алгоритмів, буде розглянуто в цьому розділі.

2.1. Розробка програмного комплексу на базі нейронної мережі типу YOLO

2.1.1. YOLOv2

Даний алгоритм на відміну від усіх інших має принцип дивитися на зображення лише один раз. Це дуже прискорює виконання роботи алгоритму. А саме, алгоритм розбиває зображення на сітку з фіксованою шириною та висотою вікна.(рис 2.8). Приклад роботи зроблений на основі зображень з інтернету, але принцип залишається тим самим.

Наступним кроком кожна клітинка намагається передбачити 5 обмежувальних «коробок». Ці «коробки» – це прямокутники, що описують об'єкт.

На даному кроці даний алгоритм дає як результат, яка ймовірність того, що в даному конкретному прямокутнику знаходиться який-небудь об’єкт. Нічого більше. Не говорить про те, який тип об’єкту, а лише те, яка ймовірність, що там є об’єкт.

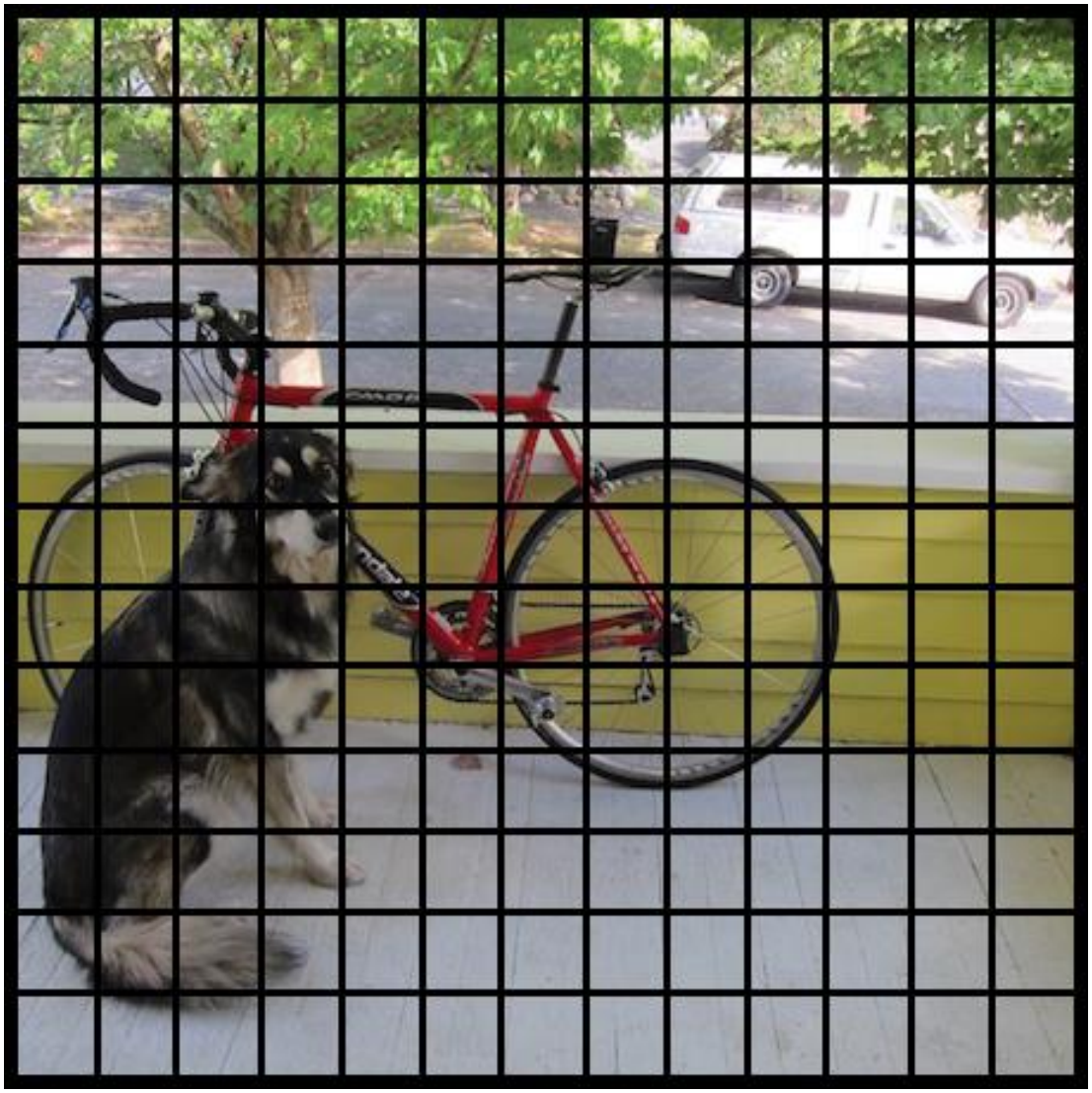


Рис2.8. Розбиття зображення на сітки

Передбачені результати можуть виглядати трохи дивно, неначе б все зображення в різного типу об’єктах (рис.2.9)

Рис2.9. Зображення з передбаченими прямокутниками. Ніж ширше краї прямокутника, тим більша ймовірність знаходження об’єкту всередині нього.

Далі для кожного прямокутника також передбачається тип об’єкту. Тобто робота алгоритму тут відбувається, як у звичайного класифікатора. У кожного прямокутника є ймовірність, того, що даний прямокутник містить в собі той чи інший об’єкт.

У випадку з даною нейронною мережею присутні всього декілька основних класів, що необхідні для даної роботи:

1. Пістолети, та ручна зброя
2. Помпова зброя
3. Ножі

Імовірність знаходження предмету в прямокутнику та імовірності конкретних класів об’єднуються, що надає нам фінальний результат, що каже про імовірність знаходження об’єкту в прямокутнику.

В описаному прикладі алгоритм на 85% впевнений, що в великому жовтому прямокутнику знаходиться пес. (рис.2.10)

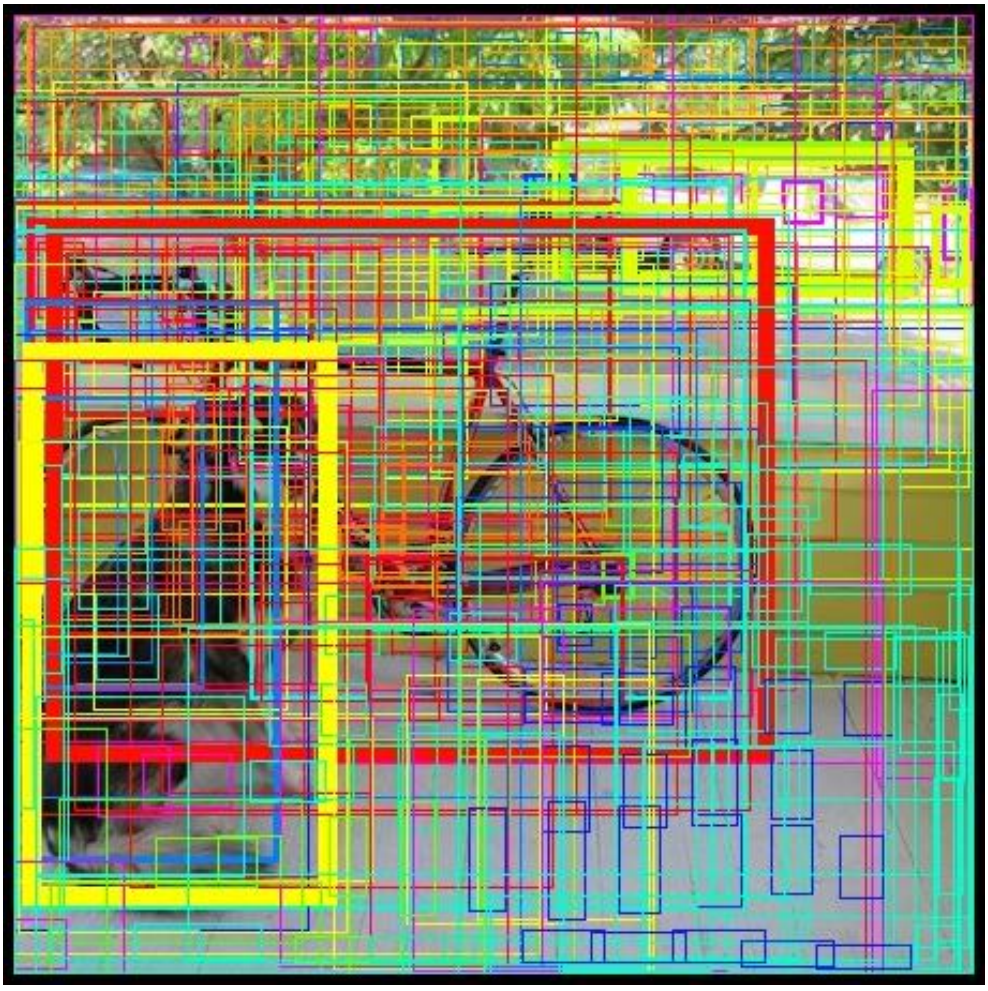


Рис.2.10. Пес знаходиться в жовтому прямокутнику

Оскільки комірок сітки 13 × 13 = 169, і кожна комірка передбачає 5 прямокутників, загалом у нас виходить 845 прямокутників. Виявляється, більшість із цих прямокутників мають дуже низькі показники достовірності, тому зберігаються лише ті коробки, кінцевий бал яких становить 30% або більше. Цей поріг можна змінювати в залежності від налаштувань. Наприклад після встановлення такого порогу на прикладі лишилося лише 3 прямокутники.

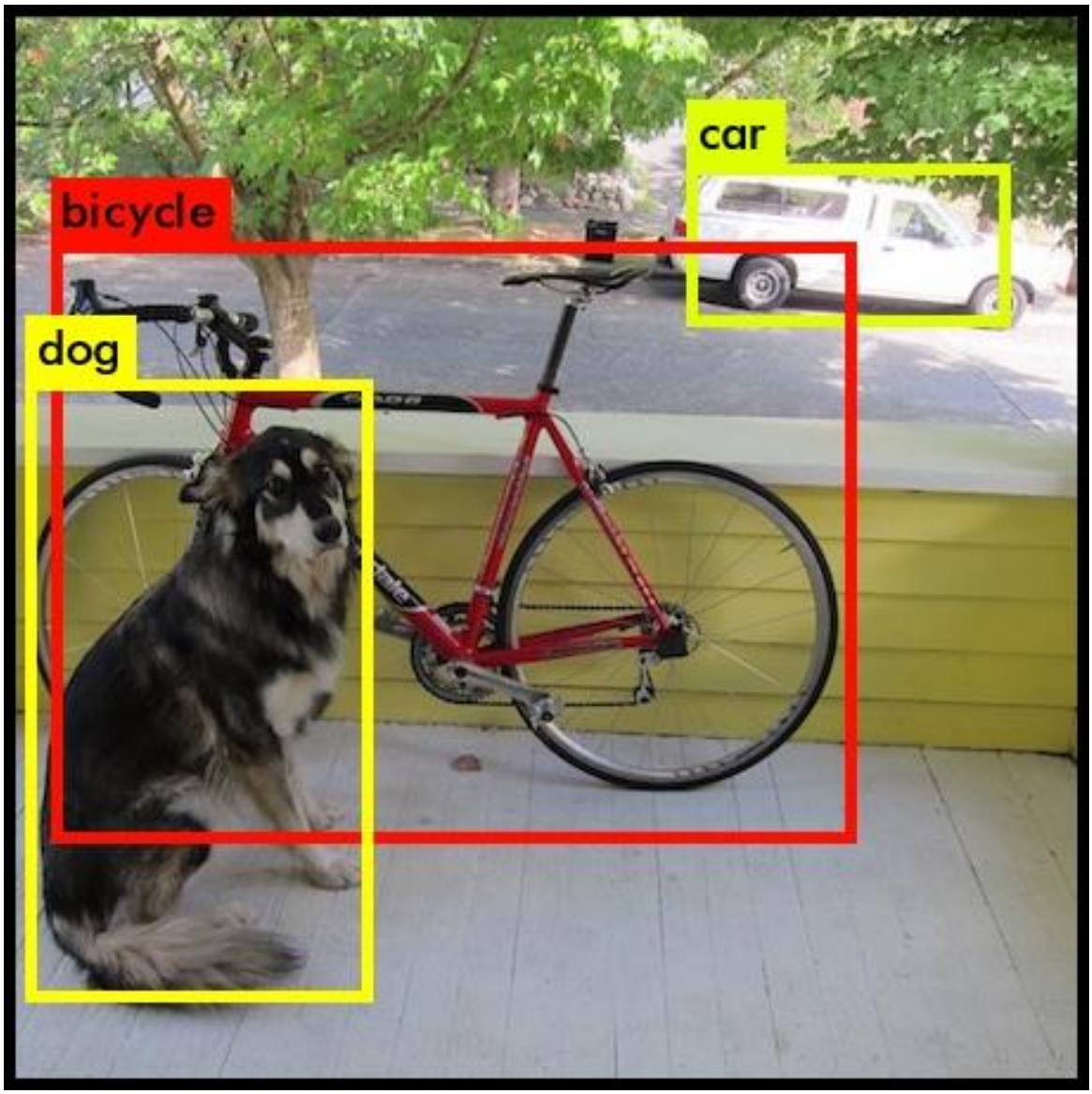


Рис.2.11. Фінальний найбільш остаточний результат

2.1.2. YOLOv3

На відміну від структури нейронної мережі типу YOLO другої версії, YOLO третьої версії працює значно більш точно. Це не єдина різниця алгоритмів. Адже такої різниці іноді можна досягати за рахунок навчання нейронної мережі. Розглянемо структуру нейронної мережі YOLO третьої версії більш детально

Швидкість була замінена на підвищення точності в YOLO третьої версії. Хоча попередній варіант працював з частотою кадрів 45 кадрів в секунду, дана

версія працює приблизно з частотою кадрів 30 кадрів в секунду. Це пов’язано зі збільшенням складності базової архітектури, що називається Darknet.

YOLO другої версії використовував власну архітектуру darknet-19, спочатку 19-шарову мережу, доповнену ще 11 шарами для виявлення об'єктів. Завдяки 30-шаровій архітектурі YOLO другої версії часто має складнощі з виявленням дрібних об’єктів. Це було пов'язано з втратою дрібнодисперсних функцій, оскільки шари зменшували вибірку вводу. Щоб виправити це, YOLO другої версії використовував відображення ідентичності, об’єднуючи карти об’єктів із попереднього шару для захоплення об’єктів низького рівня.

Однак в архітектурі YOLO другої версії все ще бракувало деяких найважливіших елементів, які зараз є основними в більшості найсучасніших алгоритмів. Немає залишкових блоків, відсутні пропускні з'єднання та відсутність вибірки. YOLO третьої версії містить усе це.

По-перше, YOLO третьої версії використовує варіант Darknet, який спочатку має 53-шарову мережу. Для завдання виявлення на нього укладено ще 53 шари, таким чином отримуємо 106-шарову повністю згорнуту базову архітектуру для YOLO третьої версії. Це причина повільності YOLO третьої версії порівняно з YOLO другої версії. На рисунку 2.12 зображено архітектуру YOLO третьої версії.

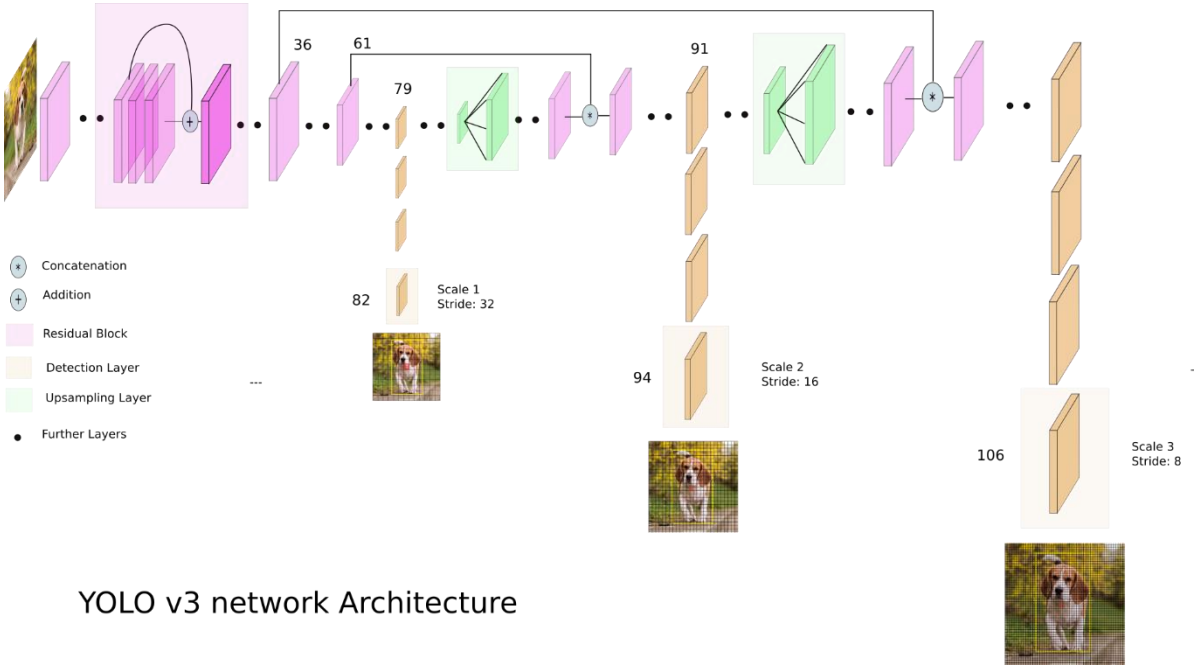


Рис. 2.12 Архітектура YOLOv3

Нова архітектура може похвалитися залишковими пропусками з'єднань та підвищеною вибіркою. Найбільш помітною особливістю v3 є те, що він здійснює виявлення в трьох різних масштабах. YOLO - це повністю згорнута мережа, і її кінцевий результат генерується шляхом застосування ядра 1 x 1 на карті об’єктів. У YOLO v3 виявлення відбувається шляхом застосування 1 x 1 фільтрів виявлення на картах об’єктів трьох різних розмірів у трьох різних місцях мережі.

Форма ядра виявлення становить 1 x 1 x (B x (5 + C)) (рис 2.13). Тут B - кількість меж, які може передбачити клітинка на карті об’єктів, „5” - це 4 атрибути обмежувального поля та одна довіра об’єкта, а C - кількість класів. У YOLO v3. Карта об'єктів, вироблена цим ядром, має однакову висоту і ширину попередньої карти функцій і має атрибути виявлення вздовж глибина, як описано вище.

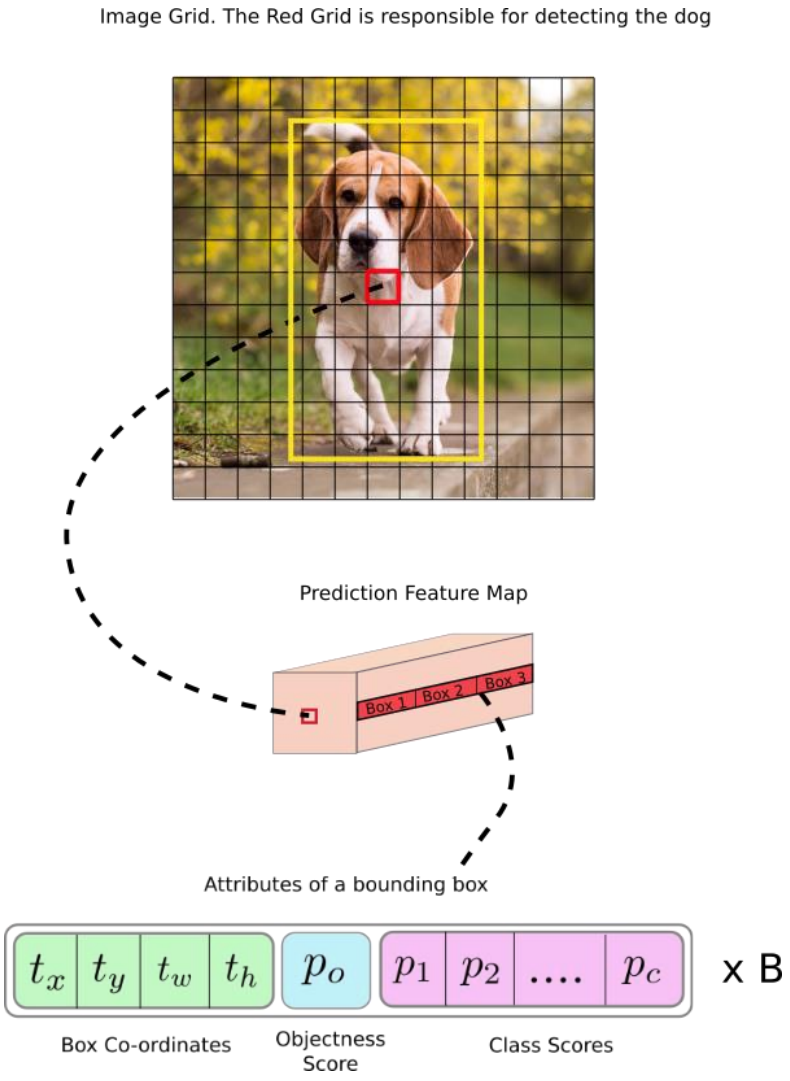


Рис. 2.13. Межі для об’єкту

В даному випадку система оброблює дані, отримані із зображення тричі. При зображенні 416 x 416 це робиться з сітками розміром 13 x 13, потім 26 x 26, потім 52 x 52.

Саме на обробку цих даних витрачається більшість обчислювальних можливостей у YOLO третьої версії, адже задля того, щоб отримати дрібні об’єкти, виконуються зайві обчислення, які підвищують точність знаходження об’єктів.

2.1.3. Власна розробка

Виходячи з суті роботи алгоритмів YOLO другої та третьої версій. Можна зробити висновок, що в даному випадку пошук небезпечних об’єктів з камер

відеоспостереження немає необхідності шукати не дрібні об’єкти, адже зброя зазвичай на зображеннях займає відносно невеликий розмір.

Таким чином було прийнято рішення змінити архітектуру методом зменшення кількості шарів нейронної мережі, що відповідають за підготовку та обробку зображення на таке низьке розширення сітки. В даному варіанті буде пропущена низка шарів, що використовуються для розбиття на сітку 13 x 13. Це також прибирає кілька шарів після обробки 13 x 13, що збільшує один раз розмірність шару нейронної мережі. Залишаються лише шари 26 x 26 та 52 x 52.

Дана структура зображена на рисунку 2.14

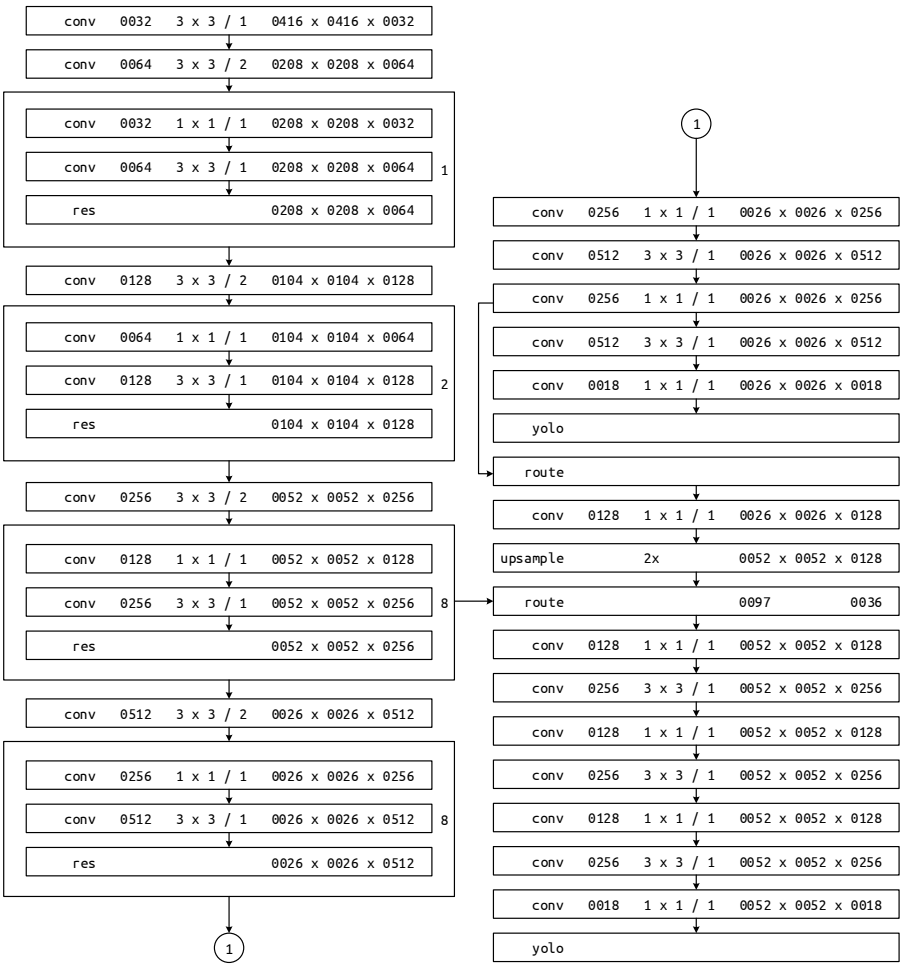


Рис 2.14. Розроблена архітектура нейронної мережі

Детальніше розглянути архітектуру можна у додатку 1.

Як можна побачити з набір даних зображень з камер відеоспостереження, зброя дійсно займає не великі розміри на зображеннях. (Рис 2.15)

Рис 2.15. Набір даних зображень з камер відеоспостереження, де є зброя.

Розглянемо більш реальний приклад, спершу працює алгоритм YOLO, який визначає рамки об’єктів, далі працює класифікатор об’єкту. На рисунку 2.14 зображено результат роботи алгоритму YOLO, який знайшов об’єкт.

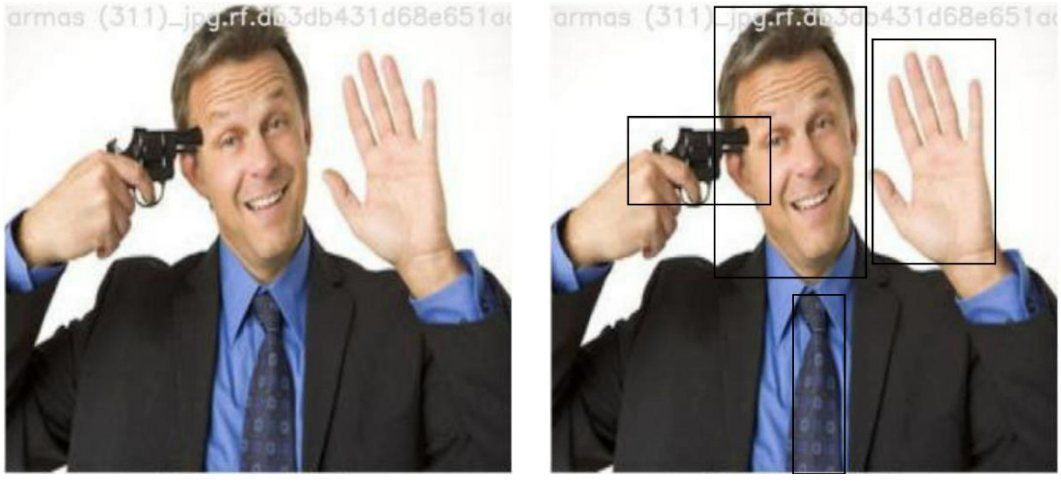


Рис.2.16. Зображення до пошуку та після пошуку об’єктів

Слід зазначити, що на рисунку прибрані прямокутники, що мають доволі низькі імовірності.

Розглянемо роботу класифікатора на основі зображення, де можна побачити пістолет.

Наступний крок – це згорткова нейронна мережа визначає за рахунок фільтру краї об’єкту. Працює це наступним чином. Зображення береться в чорно-білому спектрі, і по зображенню проковзує ядро, що генерує нове зображення зі значеннями, залежними від даного ядра. Ядро що використовується для знаходження країв містить в середині коефіцієнт 8 та навколо коефіцієнти -1. Тобто якщо з однієї сторони буде різка зміна кольору, то це означає, що в даному місці ймовірно є якась межа. Далі це все фільтрується функцією активації ReLU, для того щоб на зображенні залишилися лише контури (рис 2.15).

Рис.2.17. Зображення до обробки та після обробки фільтрами

Наступний крок – це використання фільтру Max_Pooling він дещо зменшує якість зображення, але саме це дає фільтру знайти конкретні особливості об’єкту. На рисунку 2.16 зображено до і після використання даного фільтру.

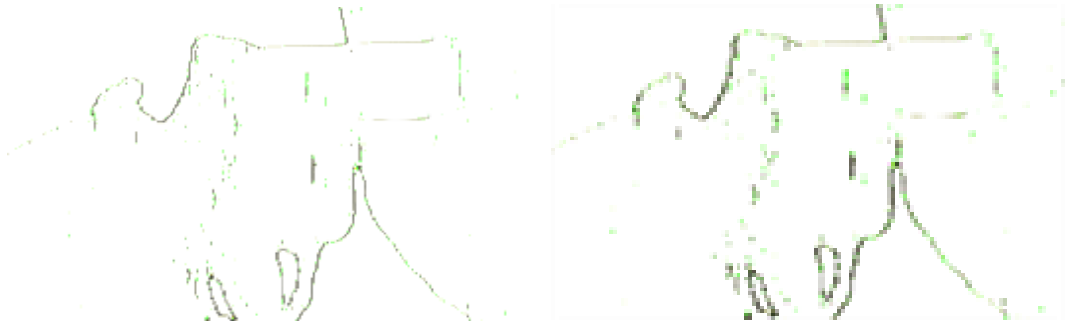


Рис. 2.18. До та після фільтру MaxPooling

Наступний крок – це розпізнавання особливостей. Цей крок тяжко зобразити, адже наступний шар згорткової мережі визначає певні особливості об’єкту. Далі йде останній шар нейронної мережі, на який поступають дані з усіх нейронів попереднього рівня, але на цьому етапі формується результат.

Після того, як об’єкт пройшов класифікацію, на виході маємо імовірність того, що це зброя, чи ні. Таким чином разом з алгоритмом YOLO маємо фінальний результат (рис 2.17). Якщо алгоритм дає результат для якогось прямокутника надто низьку ймовірність, то цей прямокутник навіть не буде зображено.

Рис 2.19. Результат роботи.

Отже таким чином працює виявлення об’єктів на зображенні. Якщо коротко, то спочатку обробка зображення, потім класифікація.

2.2. Вибір інструментів

Отже було розроблено програмний комплекс, що базується на одній з вищенаведених технологій. В цьому підрозділі будуть записані всі елементи, які були використані в процесі, особливості написання програми а також певні аспекти самої реалізації.

Для розробки було використано наступні інструменти:

1. PyTorch
2. TensorFlow
3. Jupyter Notebook

Далі буде розглянуто основні аспекти цих інструментів

2.2.1. PyTorch

Глибинне навчання дозволяє виконувати дуже широкий спектр складних завдань, таких як машинний переклад, гра в стратегічні ігри або ідентифікація об'єктів у захаращених сценах, піддаючи модель наочним прикладам. Для того, щоб зробити це на практиці, потрібні гнучкі інструменти, щоб їх можна було адаптувати до такого широкого кола проблем, щоб дозволити проводити навчання за великими обсягами даних у розумні терміни; і також потрібна навчена модель для правильної роботи за наявності мінливості вхідних даних.

PyTorch легко зрозуміти через його простоту. Багатьом працівникам легко навчатись, використовувати та налагоджувати. Хоча, як і будь-який складний домен, він має застереження та найкращі практики, використання бібліотеки, як правило, буде знайомим розробникам, які раніше використовували Python.

Більш конкретно, програмування машини глибинного навчання є дуже природним для PyTorch. PyTorch надає тип даних тензор, для зберігання чисел,

векторів, матриць або масивів загалом. Крім того, він забезпечує функції для роботи з ними. Можна програмувати з ними поступово.

Але PyTorch пропонує дві речі, які роблять його особливо актуальним для глибокого навчання: по-перше, він забезпечує прискорене обчислення з використанням графічних процесорів (GPU), часто даючи прискорення в діапазоні 50 разів порівняно з тим самим розрахунком на центральному процесорі. По-друге, PyTorch пропонує засоби, що підтримують чисельну оптимізацію загальних математичних виразів, які глибинне навчання використовує для навчання. Обидві функції корисні для наукових обчислень загалом, а не виключно для глибинного навчання. Насправді можна сміливо характеризувати PyTorch як високопродуктивну бібліотеку з підтримкою оптимізації для наукових обчислень у Python.

Драйвером дизайну для PyTorch є експресивність, що дозволяє розробнику реалізовувати складні моделі без надмірної складності, накладеної бібліотекою. PyTorch отримав широке поширення в дослідженнях, про що свідчить велика кількість цитувань на міжнародних конференціях.

PyTorch також має вагому історію про перехід від досліджень та розробок до виробництва. Спочатку він був орієнтований на робочі процеси досліджень, але PyTorch був оснащений високопродуктивним середовищем виконання C ++, який може використовуватися для розгортання моделей для виведення, не покладаючись на Python, і може використовуватися для проектування та навчання моделей на C ++. Він також розширив прив'язку до інших мов та інтерфейс для розгортання на мобільних пристроях. Ці функції дозволяють нам скористатися гнучкістю PyTorch і одночасно використовувати програми, де повний час роботи Python важко отримати або накласти дорогі накладні витрати. [40]

2.2.2. TensorFlow

TensorFlow пропонує декілька рівнів абстракції, тому він являється гнучким для різного типу потреб. Він дозволяє створювати та навчати моделі за допомогою зручного API Keras, що спрощує початок роботи з TensorFlow та машинним навчанням.

Для гнучкості є активне виконання, що забезпечує швидкий запуск та інтуїтивно зрозуміле налагодження. Для великих завдань машинного навчання можна використовувати різні стратегії розпаралелення обчислень для розподіленого навчання на різних конфігураціях апаратного забезпечення без зміни моделі.

Створення та навчання сучасних моделей можна виконувати не жертвуючи ні швидкістю ні продуктивністю. TensorFlow забезпечує гнучкість та контроль за допомогою різних API створення складних топологій, а також є можливість за допомогою активного виконання створення простих прототипів та швидкого налагодження. [41]

2.2.3. Jupyter Notebook

Jupyter Notebook – це свого роду веб записна книга, в якій можна запускати код написаний на мові Python. Даний веб-додаток містить в собі ядро, на якому виконуються всі команди. Також має розділи для звичайного тексту та для рядків коду. Кожен блок коду може виконуватися не залежно один від одного, що дає можливість в одному записникові створювати багато різних моделей, і за рахунок черги запуску елементів з кодом можна визначати, яка саме модель працюватиме в даний момент часу. Такий порядок дій досягається завдяки тому, що процес ядра може бути підключений одночасно до більш ніж одного інтерфейсу.

Ця конструкція мала на меті полегшити розробку різних інтерфейсів на основі одного і того ж ядра, але також дозволила підтримувати нові мови в тих

же інтерфейсах, розробляючи ядра на цих мовах, вдосконалення IPython, щоб зробити це більш практичним.[42]

Також Google COLAB підтримує роботу з даного записника. За допомогою сервісу Google COLAB можна виконувати обчислення на віддаленій машині, що знаходиться на серверах Google. Якщо вірно налаштувати робочий процес цього сервісу, деякі обчислення можна виконувати досить таки швидко і не залежно від обчислювальних можливостей власної машини.

В основному в роботі використовується саме цей інструмент.

2.3. Розробка інтерфейсу програми

Програма складається з одного вікна, проте для коректної роботи програми необхідні наступні компоненти:

1. Випадний список, що дозволяє обрати тип архітектури нейронної мережі, якою обробити зображення. Це зроблено в першу чергу для того, щоб можна було порівняти роботу різних типів архітектури на одній машині;
2. Кнопка для вибору обробки зображення з камери. Потрібно для того, щоб можна було перевірити роботу безпосередньо з відеокамери користувача, що під’єднана до комп’ютера;
3. Текстове поле з виводом інформації. Тут виводиться текстова інформація щодо потоку роботи програми;
4. Кнопка для відкриття зображення з файлу. Потрібно для завантаження зображення або відео з файлу. Таким чином можна перевірити швидкість роботи програми не тільки за рахунок вводу з камери, а ще й за рахунок роботи з відеопотоком;
5. Мітка, що відображає кількість кадрів за секунду. Необхідний компонент для порівняння різних архітектури;
6. Кнопка аварійного завершення. Не розроблений компонент програми, що необхідний для аварійного завершення роботи нейронної мережі, через імовірне її зависання.

```
def setupUi(self, NN):
    NN.setObjectName("NN")
    NN.resize(1400, 700)

    NN.emergency.connect(self.emergencyStop)
    NN.network.connect(self.capture)

    self.centralwidget = QtWidgets.QWidget(NN)
    self.centralwidget.setObjectName("centralwidget")

    self.pushButton_5 = QtWidgets.QPushButton(self.centralwidget)
    self.pushButton_5.setGeometry(QtCore.QRect(900, 48, 80, 60))
    font = QtGui.QFont()
    font.setFamily("Arial")
    font.setPointSize(15)
    self.pushButton_5.setFont(font)
    self.pushButton_5.setAcceptDrops(False)
    self.pushButton_5.setObjectName("pushButton_5")

    self.pushButton_4 = QtWidgets.QPushButton(self.centralwidget)
    self.pushButton_4.setGeometry(QtCore.QRect(1200, 48, 80, 60))
    font = QtGui.QFont()
    font.setFamily("Arial")
    font.setPointSize(15)
    self.pushButton_4.setFont(font)
    self.pushButton_4.setAcceptDrops(False)
    self.pushButton_4.setObjectName("pushButton_4")

    self.pushButton = QtWidgets.QPushButton(self.centralwidget)
    self.pushButton.setGeometry(QtCore.QRect(50, 590, 191, 71))
    font = QtGui.QFont()
    font.setFamily("Arial")
    font.setPointSize(17)
    self.pushButton.setFont(font)
    self.pushButton.setAcceptDrops(False)
    self.pushButton.setObjectName("pushButton")

    self.pushButton3 = QtWidgets.QPushButton(self.centralwidget)
    self.pushButton3.setGeometry(QtCore.QRect(300, 590, 250, 71))
    font = QtGui.QFont()
    font.setFamily("Arial")
    font.setPointSize(17)
    self.pushButton3.setFont(font)
    self.pushButton3.setAcceptDrops(False)
    self.pushButton3.setObjectName("pushButton3")

    self.pushButton_2 = QtWidgets.QPushButton(self.centralwidget)
    self.pushButton_2.setGeometry(QtCore.QRect(260, 110, 191, 71))
    font = QtGui.QFont()
    font.setFamily("Arial")
    font.setPointSize(17)
    self.pushButton_2.setFont(font)
    self.pushButton_2.setAcceptDrops(False)
    self.pushButton_2.setObjectName("pushButton_2")

    self.textBrowser = QtWidgets.QTextBrowser(self.centralwidget)
    self.textBrowser.setGeometry(QtCore.QRect(50, 230, 500, 280))
    font = QtGui.QFont()
    font.setFamily("Lato")
    font.setPointSize(9)
    self.textBrowser.setFont(font)
    self.textBrowser.setObjectName("textBrowser")
```

```
self.label = QtWidgets.QLabel(self.centralwidget)
self.label.setGeometry(QtCore.QRect(60, 160, 381, 61))
font = QtGui.QFont()
font.setPointSize(13)
self.label.setFont(font)
self.label.setLayoutDirection(QtCore.Qt.LeftToRight)

self.fps = QtWidgets.QLabel(self.centralwidget)
self.fps.setGeometry(QtCore.QRect(700, 50, 100, 61))
font = QtGui.QFont()
font.setPointSize(13)
self.fps.setFont(font)
self.fps.setLayoutDirection(QtCore.Qt.LeftToRight)

self.capturing = QtWidgets.QLabel(self.centralwidget)
self.capturing.setGeometry(QtCore.QRect(1000, 50, 200, 61))
font = QtGui.QFont()
font.setPointSize(13)
self.capturing.setFont(font)
self.capturing.setLayoutDirection(QtCore.Qt.LeftToRight)

self.network = QtWidgets.QLabel(self.centralwidget)
self.network.setGeometry(QtCore.QRect(800, 50, 100, 61))
font = QtGui.QFont()
font.setPointSize(13)
self.network.setFont(font)
self.network.setLayoutDirection(QtCore.Qt.LeftToRight)

self.label.setObjectName("label")
self.graphicsView = QtWidgets.QLabel(self.centralwidget)
self.graphicsView.setGeometry(QtCore.QRect(700, 120, 640, 480))
self.graphicsView.setObjectName("graphicsView")
self.comboBox = QtWidgets.QComboBox(self.centralwidget)
self.comboBox.setGeometry(QtCore.QRect(50, 120, 150, 31))
self.comboBox.setObjectName("comboBox")
self.label_2 = QtWidgets.QLabel(self.centralwidget)
self.label_2.setGeometry(QtCore.QRect(60, 60, 381, 41))
font = QtGui.QFont()
font.setPointSize(13)
self.label_2.setFont(font)
self.label_2.setLayoutDirection(QtCore.Qt.LeftToRight)
self.label_2.setObjectName("label_2")
NN.setCentralWidget(self.centralwidget)
self.menubar = QtWidgets.QMenuBar(NN)
self.menubar.setGeometry(QtCore.QRect(0, 0, 1335, 26))
self.menubar.setObjectName("menubar")
NN.setMenuBar(self.menubar)
self.statusbar = QtWidgets.QStatusBar(NN)
self.statusbar.setObjectName("statusbar")
NN.setStatusBar(self.statusbar)

font.setPointSize(12)
self.label_4 = QtWidgets.QLabel(self.centralwidget)
self.label_4.setGeometry(QtCore.QRect(490, 110, 200, 41))
self.label_4.setFont(font)
self.label_4.setLayoutDirection(QtCore.Qt.LeftToRight)
self.label_4.setAlignment(QtCore.Qt.AlignRight | QtCore.Qt.AlignVCenter)
self.label_4.setObjectName("Capture probabilities")
self.label_4.setText("Capture probabilities:")

self.retranslateUi(NN)
QtCore.QMetaObject.connectSlotsByName(NN)
```

```
self.commands = Commands()
self.commands.signals.print.connect(self.print)
self.commands.signals.lock.connect(self.unlock)
self.commands.ble.signals.print.connect(self.print)
self.commands.ble.signals.lock.connect(self.unlock)

self.connectedToC = False
self.connectionInProgress = False
self.pushButton.clicked.connect(self.connect)
self.pushButton3.clicked.connect(self.emergencyStop)
self.pushButton_4.clicked.connect(self.capture)

self.gesturesLabels = []

font = QtGui.QFont()
font.setPointSize(10)

self.print("Application started")
self.comboBox.addItem("None", 1)
offset = 30
i = 0
for val in self.commands.commands:
    self.comboBox.addItem(val.name, val.value)
    label_2 = QtWidgets.QLabel(self.centralwidget)
    label_2.setGeometry(QtCore.QRect(560, 150 + i * offset, 130, 41))
    label_2.setFont(font)
    label_2.setLayoutDirection(QtCore.Qt.LeftToRight)
    label_2.setAlignment(QtCore.Qt.AlignRight | QtCore.Qt.AlignVCenter)
    label_2.setObjectName(val.name)
    label_2.setText(val.name)
    self.gesturesLabels.append(label_2)
    i += 1

self.pushButton_2.clicked.connect(self.sendWrap)
self.pushButton_5.clicked.connect(self.onNetwork)

self.recognition = GestureRecognition(self.showFps)
self.recognition.setTerminationEnabled(True)
self.recognition.signals.result.connect(self.getRecognitionResult)

self.finished = self.recognition.signals.finished
self.gesturesCapturing = False
self.netowkrProcess = True
self.recognition.start()
self.lastCompute = None
self.frames = 0
self.time = time()
```

Після розробки інтерфейсу перейдемо до опису роботи програми та системи, що буде розглянуто в наступному розділі. Інтерфейс розроблений таким чином, щоб було зручно користуватися кінцевому користувачеві, а також таким чином, щоб було зручно провести дослідження.

Висновки до розділу 2

У розділі 2 було розглянуто технології за допомогою яких розробка даної системи являється можливою. Виходячи з цього, можна здогадатися, що технології зроблені не ідеальними людьми будуть не ідеальними, і у кожної з технологій будуть свої переваги та недоліки. Найбільша перевага TensorFlow являється в тому, що це інструмент, який розроблений безпосередньо для роботи з векторними обчисленнями на різного типу процесорах. Таким чином це досить таки добре підходить для роботи з нейронними мережами. Недоліком системи є її розмір, хоча для даного випадку це не дуже великий недолік. Використання записника Jupyter в даному випадку являється дуже корисним, адже також за допомогою Google Colab можна виконувати різні обчислення на потужних обчислювальних машинах Google.

Також що дуже важливо, в цьому розділі було розглянуто архітектури нейронних мереж YOLO другої та третьої версії, що себе показали в області ідентифікації об’єктів дуже добре. Однак один із алгоритмів виявився швидшим, другий ефективнішим. В даному розділі було прийнято рішення поєднати найліпші характеристики обох алгоритмів, та застосувати їх до конкретної задачі, та створити власний алгоритм.

В наступному розділі буде розглянуто запуск програми та порівняння цих двох алгоритмів з власною розробкою.

РОЗДІЛ 3
ПРИКЛАДНІ АСПЕКТИ ЗАСТОСУВАННЯ ПРОГРАМНОГО КОМПЛЕКСУ ІДЕНТИФІКАЦІЇ НЕШТАТНИХ СИТУАЦІЙ НА ОБ'ЄКТАХ КРИТИЧНОЇ ІНФРАСТРУКТУРИ

В даному розділі розглянуто роботу програми а також методику того, як слід запускати та навчати нейронну мережу.

3.1. Інструкція користувача

Як вже було описано у розділі 2 програма має декілька основних компонентів, та може працювати з YOLOv2, YOLOv3 та власною розробкою.

По перше, для запуску програми необхідно запустити її за допомогою Python3. Тобто для того, щоб програма працювала необхідно встановити Python3, пакети для PyTorch, Tensorflow та PyQt.

На рисунку 3.1 наглядно показаний інтерфейс програми.

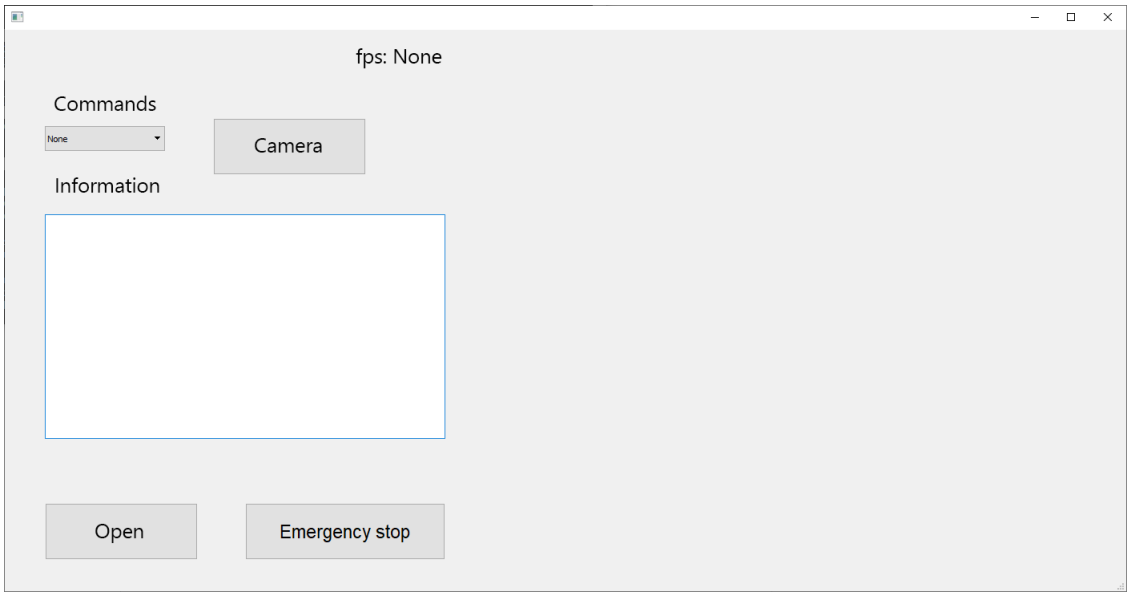


Рис. 3.1. Інтерфейс програми

У випадному списку є три опції None, v2, v3, що відповідають за власну розробку, YOLO другої версії та YOLO третьої версії відповідно.

Для того щоб відкрити зображення з файлу слід натиснути кнопку Open, тоді відкриється діалогове вікно, що запропонує обрати файл. (рис 3.2)

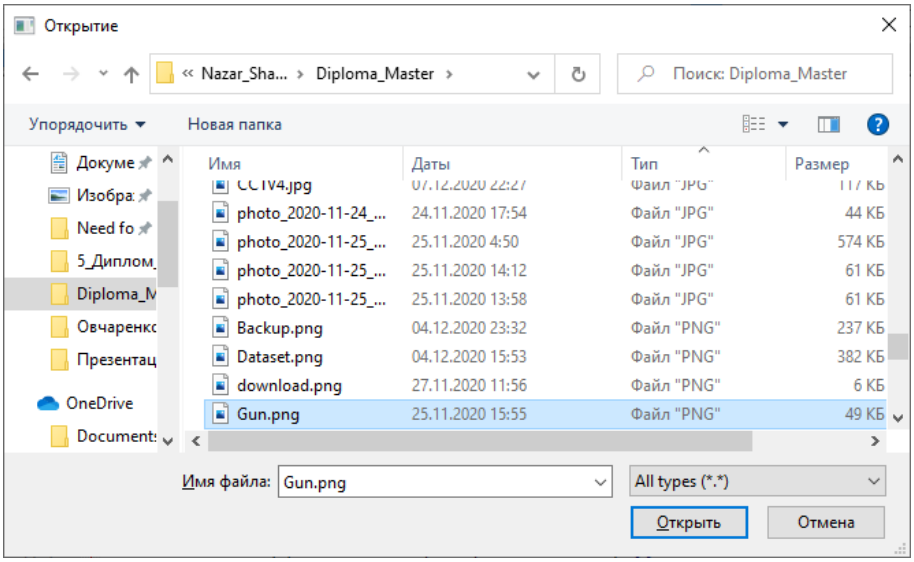


Рис 3.2. Діалогове вікно для вибору файлу

який необхідно відкрити і тоді програма обробить зображення за допомогою обраної системи. (рис 3.2)

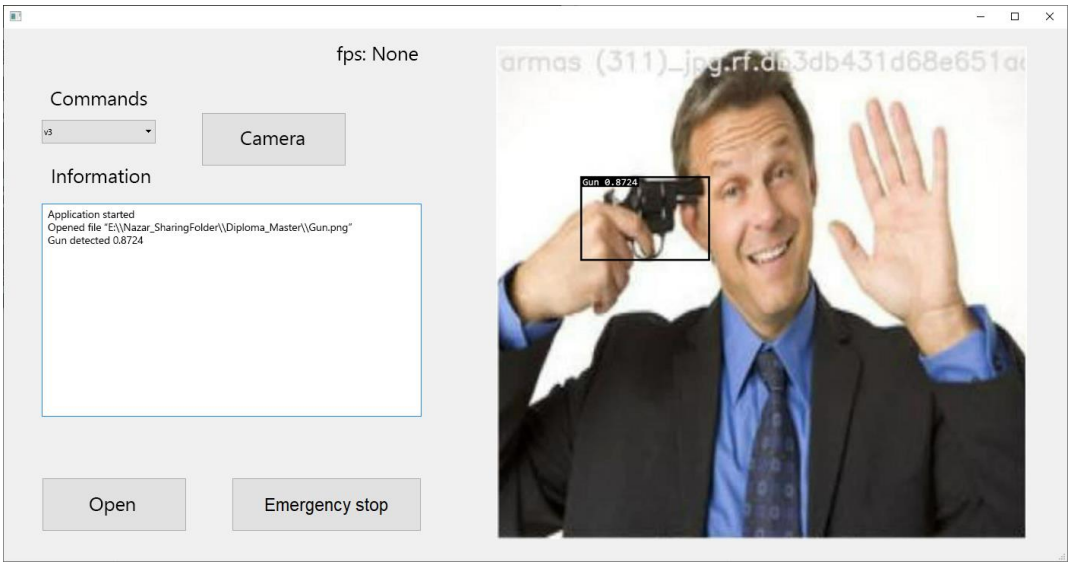


Рис 3.2. Зображення оброблене за допомогою YOLO третьої версії

За допомогою кнопки Open також можна відкривати файли, що мають відео формат. (рис 3.3)

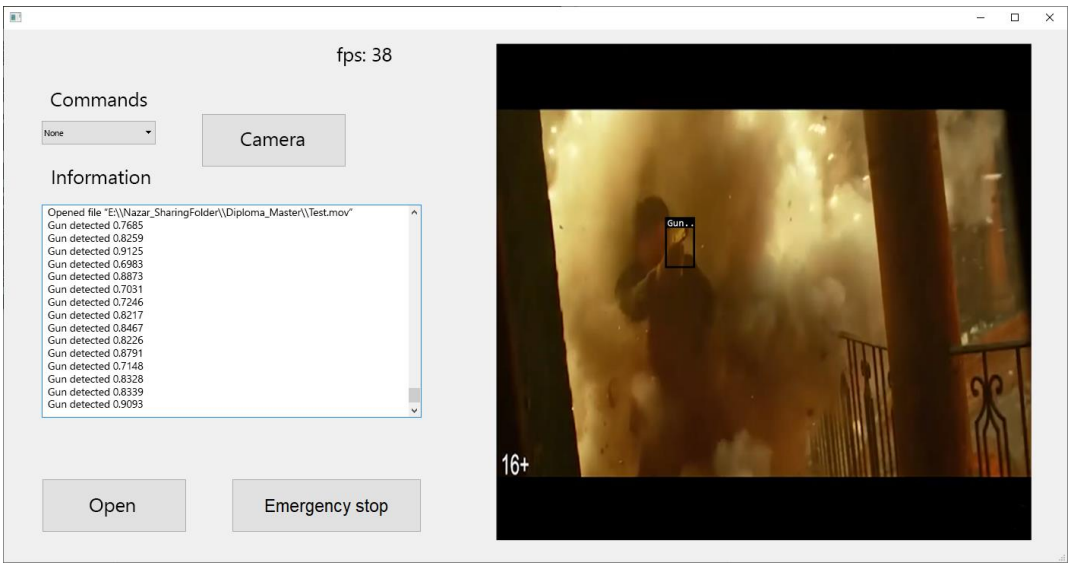


Рис. 3.3. Відкритий файл, що має відеоформат.

При натисканні на кнопку Camera можна побачити, що починається в поле зображення транслюватися відео, що записує веб-камера, що під’єднана до комп’ютера (Рис. 3.4.). Нажаль на даному етапі не розроблено логіки вибору веб камери, а обирається перша, яка ідентифікується. Якщо камери немає, то програма може зависнути.

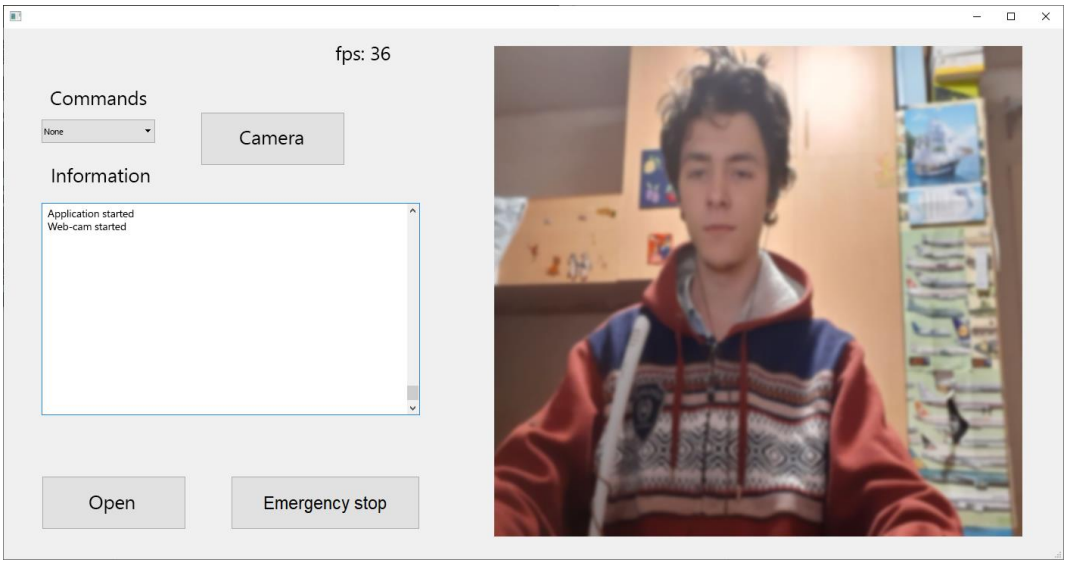


Рис. 3.4. Демонстрація роботи камери

3.1. Інструкція налаштування системи

3.1.1. Попередня підготовка

Якщо на вашому локальному комп’ютері немає графічного процесора (графічний процесор) або ваш графічний процесор не підтримується CUDA (перевірте сумісність вашого графічного процесора з CUDA тут), цей процес може бути значно болючим. На щастя, замість того, щоб витратити сотні євро на придбання та встановлення графічного процесора, ми можемо навчити YOLOv3 безкоштовному, зручному та простому у використанні інструменту дослідження машинного навчання - GOOGLE COLAB. Специфікація обладнання, надана в середовищі GOOGLE COLAB, є (джерелом):

- Графічний процесор: 1xTesla K80, обчислювальний 3.7, маючи 2496 ядер CUDA, 12 ГБ GDDR5 VRAM (найголовніше, він сумісний з CUDA)
- Процесор: 1ходноядерні гіперпотокові процесори Xeon @ 2,3 ГГц, тобто (1 ядро, 2 потоки)
- Оперативна пам’ять: ~ 12,6 ГБ
- Диск: ~ 33 ГБ в наявності

3.1.2. Підготовка даних

Набір даних, необхідний для підготовки детектора за допомогою YOLOv3, містить 2 компоненти: зображення та мітки. Кожне зображення буде асоційоване з файлом міток (як правило, текстовим файлом), який визначає клас об'єкта та координати об'єкта на зображенні з наступним синтаксисом: <object-class> <x_center> <y_center> <width> <height>

Для створення маркувань із зображень рекомендую використовувати графічний інтерфейс для позначення об'єктів та створення файлів маркувань, таких як Yolo_label, OpenLabeling, Yolo_mark, BBox-Label-Tool тощо (рис. 3.5). Це значно полегшить вам життя оскільки вам потрібно лише перетягнути обмежувальне поле навколо вашого об’єкта на зображенні, і програмне забезпечення автоматично створить файл міток.

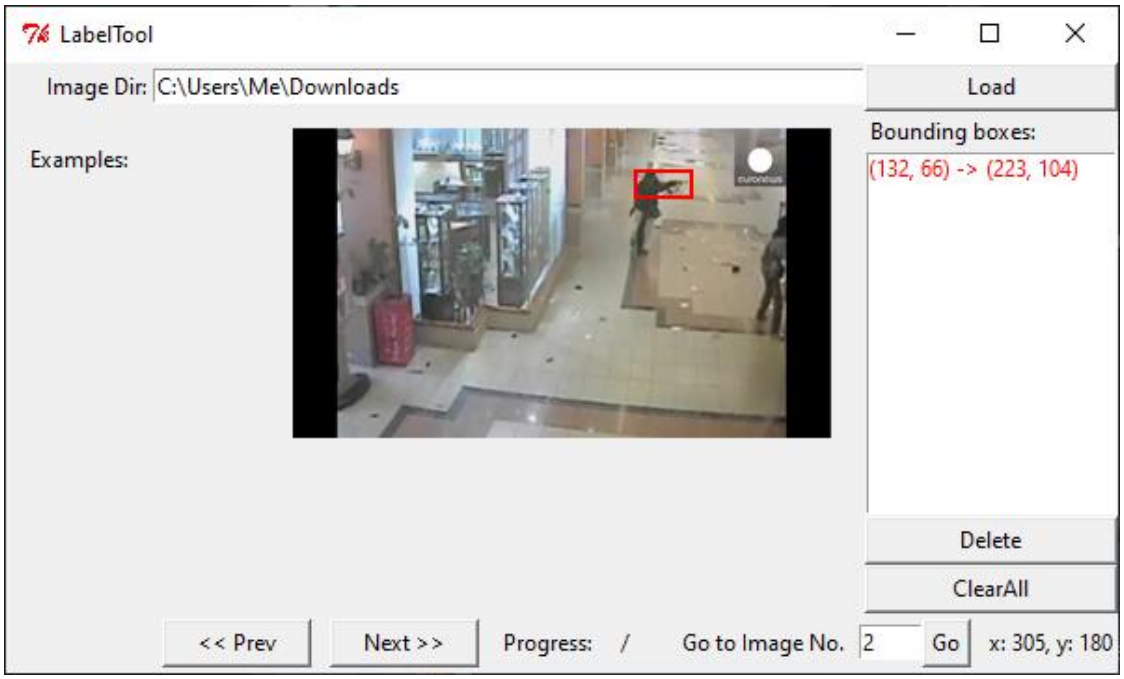


Рис.3.5. Позначення вікон, що обмежують об'єкти, за допомогою BBox-Label-Tool (джерело)

Найліпше використовувати набір даних зображень саме з вашої камери відеоспостереження, або з інших камер відеоспостереження, адже нас цікавлять саме реальні результати. Але так як ми проводимо в наукових цілях роботу, то було завантажено звичайні дані (рис. 3.6).

Очевидно, що ніж більше даних, тим ліпше буде навчена нейронна мережа.

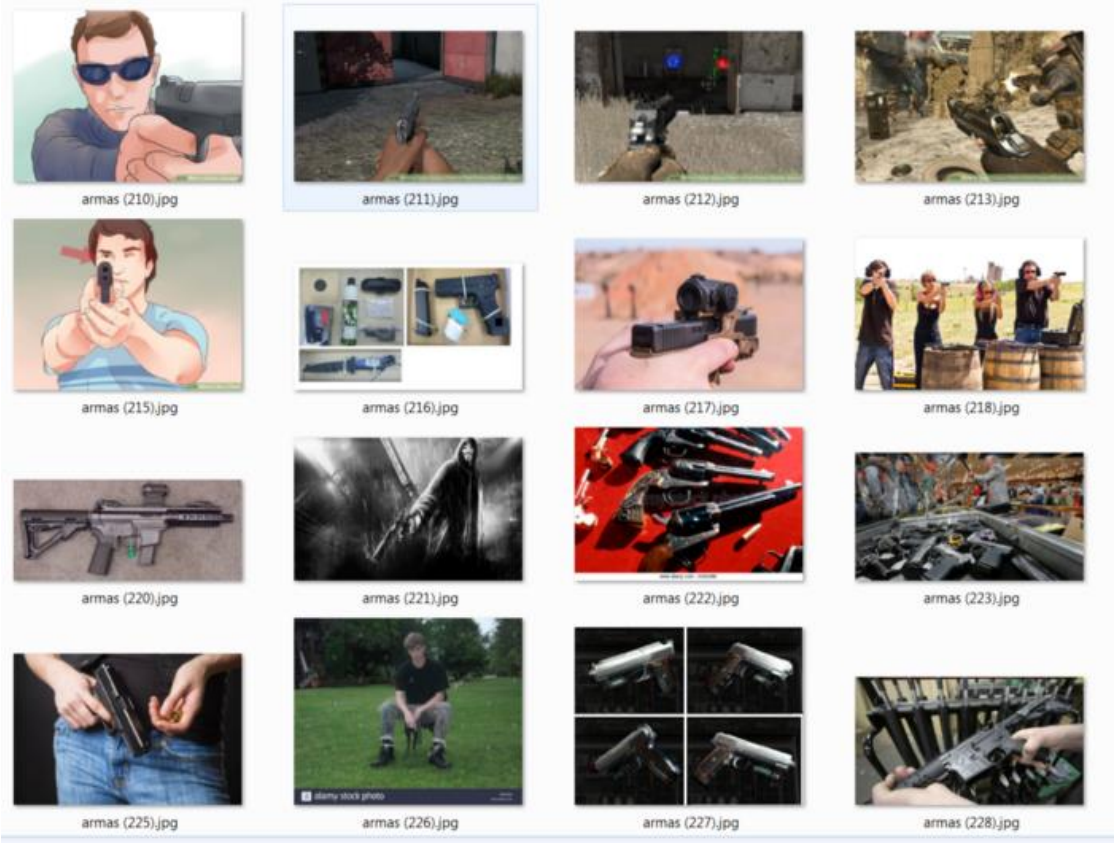


Рис.3.6. Набір даних ручної вогнепальної зброї

3.1.3. Darknet

YOLOv3 створений на Darknet, нейронній мережі з відкритим кодом для навчання. Щоб завантажити Darknet скористуємось Git:

```
cd ~
git clone https://github.com/pjreddie/darknet
```

Далі треба відкрити файл Makefile у папці Darknet. Якщо операційна система Windows, можна відкрити Makefile за допомогою Блокнота. Якщо операційна система Ubuntu, слід відкрити термінал всередині папки Darknet і введіть (ви також можете відкрити Makefile з будь-яким доступним IDE, встановленим у вашій системі):

```
gedit Makefile
```

Далі в Makefile змініть GPU = 1, CUDNN = 1 і OPENCV = 1. Потім завантажте попередньо навчену модель Darknet на Imagenet безпосередньо звідси (154 Мб). Для Ubuntu ви можете завантажити його з терміналу, як показано нижче:

```
cd darknet wget https://pjreddie.com/media/files/darknet53.con
v.74
```

Нарешті, скопіюйте та перезапишіть папку даних (включаючи папки зображень та міток) із набору даних Handgun у папку darknet.

3.1.4. Змінити файл конфігурації

У каталозі darknet \ cfg створіть копію “yolov3.cfg” в тій же папці та перейменуйте її на “yolov3_custom_train.cfg”. Відредагуйте файл, як показано в інструкції нижче (або завантажте його звідси), щоб він відповідав середовищу COLAB:

```
# Line 8 & 9:
width = 416, height = 416 # Line 20
max_batches = 6000 # Line 22
steps = 5400 #Line 603, 689, 776:
filters = 18#Line 610, 696, 783:
classes = 2
```

Тут ми застосовуємо таку формулу: (фільтри = (класи + 5) * 3) для обчислення кількості фільтрів. Наприклад, в даному випадку потрібно виявити 3 об’єкти, тоді фільтр матиме значення: filters = (3 + 5) * 3 = 24 і classes = 3. У каталозі darknet\examples папки слід відкрити файл “detector.c”. У рядку 138 слід замінити цей рядок, як показано нижче:

```
if(i%1000==0 || (i < 1000 && i%100 == 0)){
```

Ця модифікація повідомляє вашій програмі зберігати вагу вашої моделі кожні 100 епох, якщо кількість епох менше 1000, з іншого боку, зберігайте вагу кожні 1000 епох. Це має вирішальне значення, оскільки COLAB дозволяє проводити лише 12 годин тренувань із відкритим браузером, тому важливо зберегти вагу на своєму Диску до того, як сервер буде вимкнено, а всі файли в каталозі виконання будуть видалені. Пізніше ви все ще зможете завантажити збережену попередньо навчену вагу і знову навчити модель.

3.1.5. Розділення набору даних

Щоб уникнути надмірного встановлення та досягнення об’єктивної оцінки щодо моделі, нам потрібно розділити набір даних на навчальний набір та набір для перевірки роботи нейронної мережі. Залежно від кількості зображень у наборі даних, який можна перевірити набором перевірки, приблизно від 5% (малий набір даних) до 30% (великий набір даних) від загального набору даних. Ми використаємо файл „train.txt” та „val.txt”, щоб визначити шлях до каталогу навчальних та перевірочних зображень, як показано нижче:

```
data/images/armas (2976).jpg
data/images/armas (419).jpg
data/images/armas (1416).jpg
data/images/armas (1591).jpg
data/images/armas (1227).jpg
data/images/armas (1509).jpg
data/images/armas (407).jpg
data/images/armas (2956).jpg
data/images/armas (2843).jpg
data/images/armas (1510).jpg
```

На наступному кроці нам потрібно створити файл “yolo.names” у каталозі darknet\data з іменами об’єктів у кожному новому рядку. Наприклад, у даному випадку нам потрібно виявляти лише пістолет та ніж, тому перший рядок повинен бути "gun" а другий відповідно "knife".

Нарешті, після цього слід створити файл “yolo.data” у каталозі darknet\data, що містить (файл можна завантажити звідси):

```
classes= 1 #number of objects, in our case is 1
train  = data/train.txt
valid  = data/val.txt
names = data/yolo.names
backup = backup
```

3.1.6. Налаштуйте середовище COLAB

Для того, щоб тренуватися на COLAB, спочатку потрібно завантажити набір даних та папку darknet на Google Drive. Я припустив, що ви знайомі з Google Drive, якщо ні, ви можете звернутися до цього посібника. Тепер ваш каталог darknet повинен містити:

- папка images в каталозі darknet\data містить 3000 зображень
- папка ярликів у каталозі darknet\data містить 3000 .txt файл
- train.txt, val.txt, yolo.data, yolo.names в каталозі darknet\data
- yolov3_custom_train.cfg у каталозі darknet\cfg

Потім слід заархівувати папку darknet і завантажити її на свій Google Drive також слід переконатися, що ваш файл має формат darknet.zip. Створіть нову папку на своєму Диску та назвіть її “резервна”. Потім слід перейти на сторінку https://colab.research.google.com/ і увійти за допомогою свого облікового запису Google. Далі слід обрати File -> New Python 3 notebook (рис.3.7)

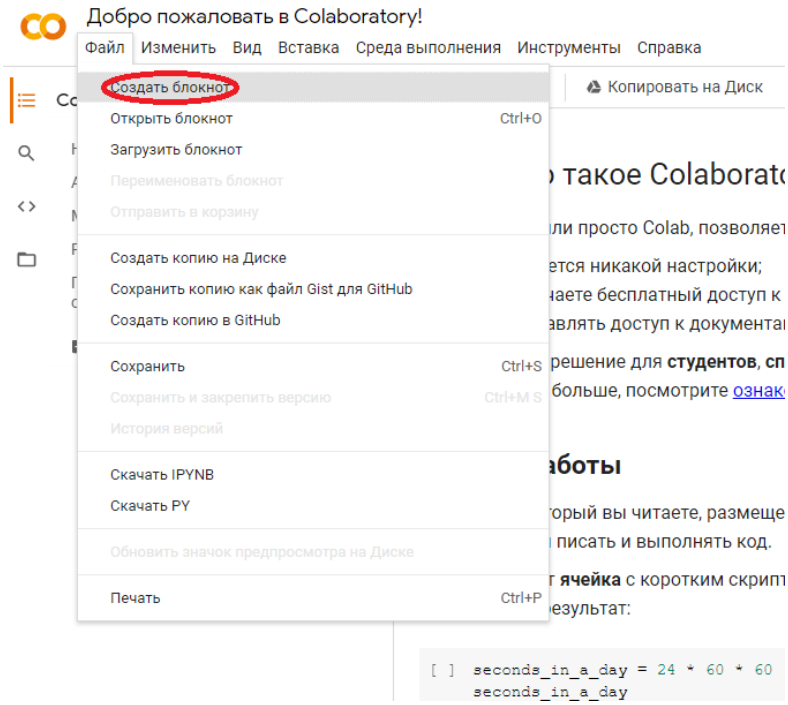


Рис.3.7. Створення нового блокнот на COLAB

Щоб увімкнути графічний процесор, слід обрати Runtime -> Change runtime type, в Hardware accelerator виберіть GPU (рис. 3.8)

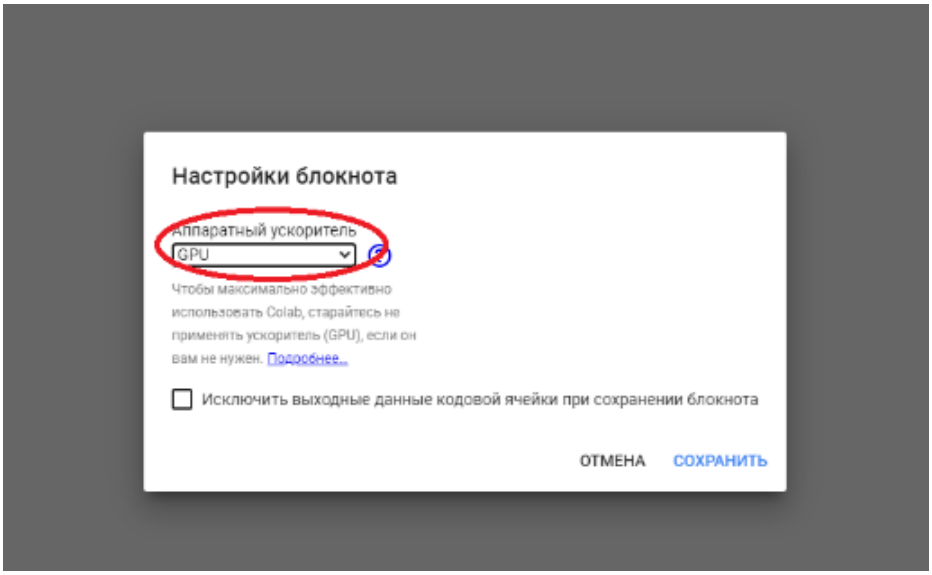


Рис.3.8. Увімкнення графічного процесору графічний процесор у вашому середовищі COLAB

3.1.7. Тренування за допомогою COLAB

У зошиті нижче наведено покрокову процедуру навчання та пояснення кожної комірки коду. Будь ласка, переконайтеся, що ви виконуєте кожну комірку по порядку, натискаючи піктограму відтворення ліворуч або натискаючи Shift + Enter і читаючи інструкцію перед запуском блокнота.

Детальніше з лістингом можна ознайомитися у Додатку 2

3.1.8. Прогнозування за допомогою YOLOv3 за допомогою OpenCV

Отримавши натреновані ваги, далі слід використовувати OpenCV для завантаження архітектури YOLOv3 і використовувати ці ваги для прогнозування.

COLAB NOTE:

- Підключіть COLAB до Google Drive (рис.3.5)

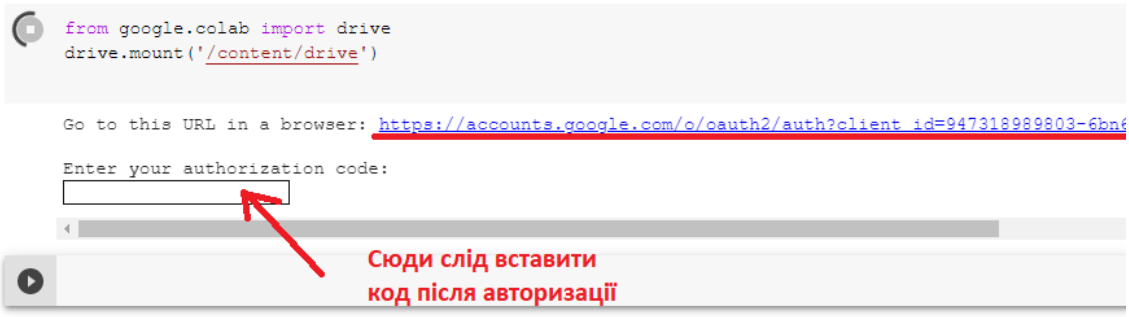


Рис.3.9. Підключення записника COLAB до диску.

- Код авторизації має бути приблизно таким як вказано на рисунку 3.6.

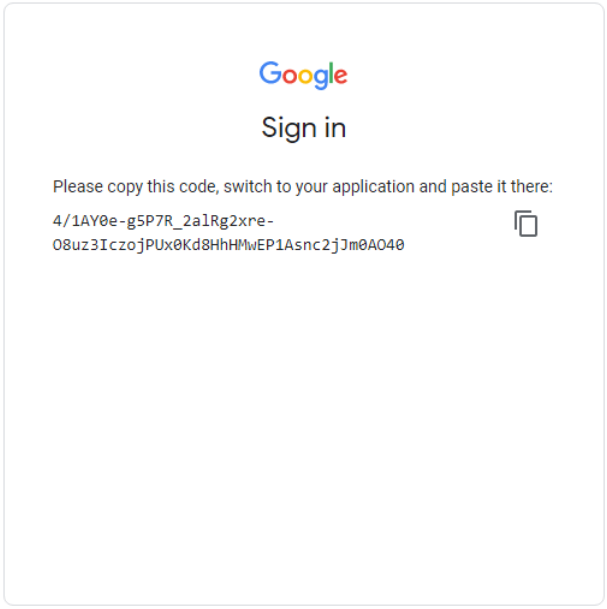


Рис.3.9. Ключ від Google

- Завантажити набір даних і папку darknet у робочий простір COLAB, розпакувавши файл “darknet.zip” (рис.3.10).

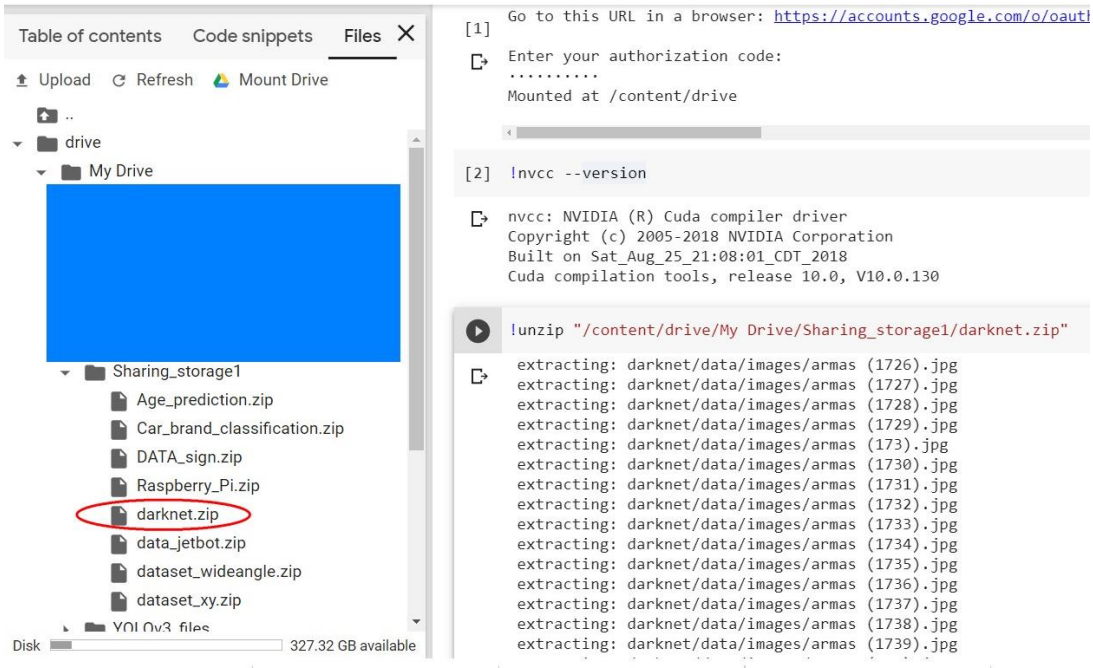


Рис.3.10. Доступ до директорії з COLAB

- Синтаксис мусить бути таким ! Unzip + <path_to_darknet.zip_in_your_GDrive> , щоб витягти папку darknet. У моєму випадку "darknet.zip" знаходиться в розділі "Мій диск"/"Sharing_storage1"
- Збереження ваги під час тренувань на своєму Google Drive

Цей крок важливий, оскільки середовище COLAB буде вимикається через 12 годин, а всі файли, що знаходяться в робочому просторі, видаляються. Тут ми визначаємо символічне посилання для збереження ваги безпосередньо в папці резервної копії, яку створено в GDrive раніше. У моєму випадку моїм каталогом папок резервних копій є My Drive/YOLOv3_weight/ backup (рис.3.8).

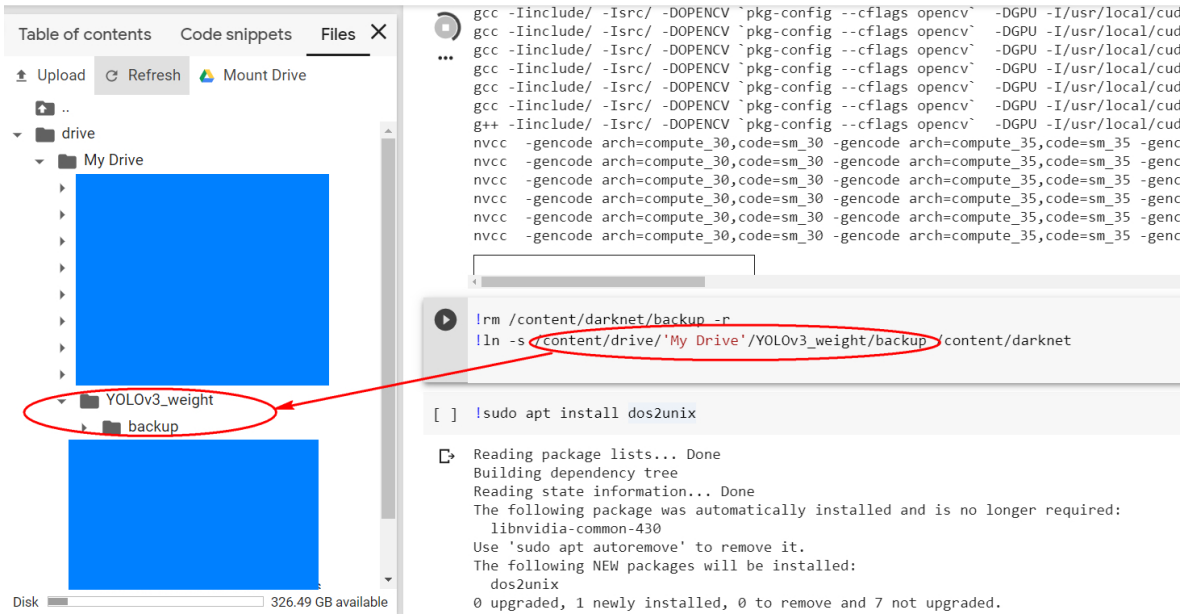


Рис.3.8. Збережені ваги під час тестувань.

Коли система достатньо натренована, можна знайти на своєму диску ваги в папці, в якій зберігаються, ці ваги можна копіювати з диску та вставити до програми. Тоді програма буде ідентифікувати все за рахунок власно натренованими вагами. Завантажувати слід замість файлу з назвою “yolov3_900.weights”.

3.2. Порівняльні характеристики

Як було сказано раніше, для роботи використано було декілька існуючих систем, та розроблена система. В цьому підрозділі наведено порівняльну характеристику даних систем

Таблиця 3.1. Порівняльна характеристика різних систем ідентифікації нештатних ситуацій

Алгоритм	Виявлення типів	Алгоритм	mAP	Переваги	Недоліки
Комп'ютерна система бачення для візуального виявлення пістолета на основі комп'ютерного бачення з використанням детектора точки Харріса [43]	Ручна зброя (пістолети)	Кольорова сегментація та детектор точок інтересу Харріса	~84%	Хороша точність на думку автора	Більше помилкових спрацьовувань і швидкості не згадується, а також не використовується стандартний набір зображень
Візуальне виявлення ножів у захисних програмах за допомогою моделей Active Appearance [44]	Ножі	Детектор кутів Харріса	~92%	Хороша точність при відсутності хибних позитивних спрацьовуваннях	Підходить лише для рентгенівських знімків. Буде повільним та точним у реальному часі.
Методи виявлення безпеки людини від прихованої зброї [45]	Ручна зброя (пістолети)	Використання обробки зображень за допомогою інфрачервоних променів та	Не дано	Без переваг	відсутність точності та необхідність рентгенівських знімків

		сенсорної технології			
Сигналізація автоматичного виявлення пістолета у відеозаписах за допомогою глибокого навчання [46]	Ручна зброя (пістолети)	Faster R-CNN	~84%	Висока кількість вірних позитивних спрацьовувань	Забагато помилкових спрацьовувань, які в деяких тестових наборах даних знижують точність до 53%, а продуктивність становить 0,2 секунди на кадр
Розробка класифікатора виявлення зброї в режимі реального часу [47]	Ручна зброя (пістолети)	OverFeat	~89%	Висока точність	Дуже повільний 1.3 секунди на кадр
Виявлення ручного пістолета за допомогою швидшого глибокого навчання R-CNN [48]	Ручна зброя (пістолети)	Faster R-CNN	~93%	Висока точність згідно з автором	Не надано інформації щодо швидкості
Прогнозування місця злочину шляхом виявлення загрозливих об’єктів за допомогою згорткової нейронної мережі [49]	Ручна зброя, ножі, кров	CNN	~90%	Висока точність	Не надано інформації щодо швидкості
Терагерцове виявлення зображення за допомогою вдосконаленої прискореної згорткової нейронної мережі[50]	Ручна зброя (пістолети)	R-CNN YOLOv2	~70%		

Система, що базується на YOLOv2	Ручна зброя	YOLOv2	~69%	Відносно низька точність	Висока швидкість роботи
Система, що базується на YOLOv3	Ручна зброя	YOLOv3	~91%	Відносно висока точність	Низька швидкість порівняно з YOLOv2
Власна система	Ручна зброя, ножі +	DarkNet YOLOv3	~88%	Відносно висока точність в порівнянні з YOLOv2 Швидкість не значно більша за YOLOv3	Швидкість менша ніж YOLOv2

Висновки до розділу 3

У розділі 3 було реалізовано програмний комплекс ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури. Основними технологіями для розробки були сервіс від Google – Google COLAB, та реалізація нейронної мережі на основі архітектури власної архітектури, що являється модернізованою архітектурою YOLOv3. В ході реалізації було використано технології, що були обрані в Розділі 2.

Найбільш затратним зі сторони апаратних ресурсів та часу були завдання навчання та нейронної мережі із власними цілями. Оскільки слід було прогнати тестові дані понад 1000 епох. Також деякі дані слід було підготувати перед тим, як запускати на навчання нейронної мережі.

В даному розділі також наявна детальна інструкція користувача щодо запуску програмного продукту та повна інструкція користувача для взаємодії з розробленою системою.

Розроблена система має наступні можливості. Навчання системи на власних тестових даних з використанням попередньо натренованої нейронної мережі, що має сенс для збільшення точності ідентифікації в конкретному місці за допомогою конкретної камери відеоспостереження. А також ідентифікація нештатних ситуацій із поданих відео матеріалів, фото матеріалів а також із веб-камери.

Також в цьому розділі було зроблено висновок, що власна розробка має сенс, адже справляється із задачею достатньо добре. Також зроблено висновок, що за допомогою власної розробки тяжко ідентифікувати великі об’єкти, але в даному випадку це не являється великою проблемою, адже на камерах відеоспостереження зброя не часто являється великими об’єктами.

РОЗДІЛ 4. РОЗРОБКА СТАРТАП ПРОЕКТУ

4.1 Інформаційна карта проекту.

Таблиця 4.1 Інформаційна карта проекту

1. Назва проекту	Програмний комплекс ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури.
2. Автори проекту	Юрченко Назар
3. Коротка анотація	Даний проект дозволяє використовувати програмний комплекс для того, щоб ідентифікувати нештатні ситуації на об’єктах критичної інфраструктури. Його можна використовувати для того, щоб інсталювати в комп’ютери персоналу, що відповідає за безпеку на об’єкті. Це може бути необхідним щоб не пропускати критичні ситуації за рахунок людського фактору.
4. Опис проблеми, яку вирішує проект	В сучасному світі спостерігається досить таки велика кількість правопорушень. Деякі з них не критичні, а деякі з них дуже важливі. Тому для того, щоб запобігати, і вчасно реагувати на такі дії цілком може бути потрібна така розробка
5. Головні цілі та завдання проекту	Головна ціль: Розробити зручний у використанні додаток, що забезпечить надійність та швидкість у розпізнаванні злочинів. Завдання: Підвищити рівень безпеки на об’єктах критичної інфраструктури..
6. Очікувані результати	В результаті виконання проекту буде забезпечено вищий рівень безпеки, на об’єктах критичної інфраструктури, підвищений рівень реакції на правопорушення.

4.2 Організація роботи стартапу

Таблиця 4.2 Склад команди та ролі учасників

	Спеціальність	**Роль**
1.	ІТ-спеціаліст	Генератор ідеї
2.	ІТ-спеціаліст	Спеціаліст
3.	Менеджер	Координатор
4.	Маркетолог	Дипломат

Таблиця 4.3 Поставлені завдання та час на їх виконання

	Завдання	**Час**
1.	Дослідження методів реалізації	0,5
2.	Досліження методів взаємодії команди	0.5
3.	Розподіл обов’язків	0,5
4.	Видача ТЗ	0,5
5.	Розробка ПЗ	6
6.	Тестування ПЗ	3
7.	Пошук інвесторів	2
8.	Юридичне забезпечення	2
9.	Просування товару	2
10.	Рекламна кампанія	2
11.	Аналітика	1 + 1 +1

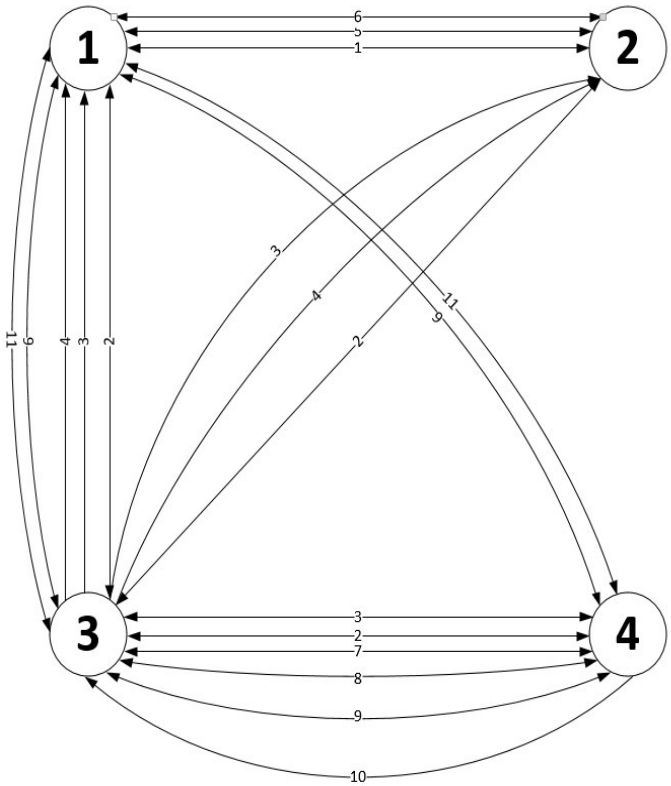


Рис. 4.1 Граф розподілення завдань між учасниками

Розрахунок загруженості кожного з учасників Загруженість = Вхід / Вихід

Баланс: 0,7 – 1,3

1 = 11 / 8 = 1,375

2 = 6 / 4 = 1,5

3 = 10 / 13 = 0,77

4 = 7 / 8 = 0.875

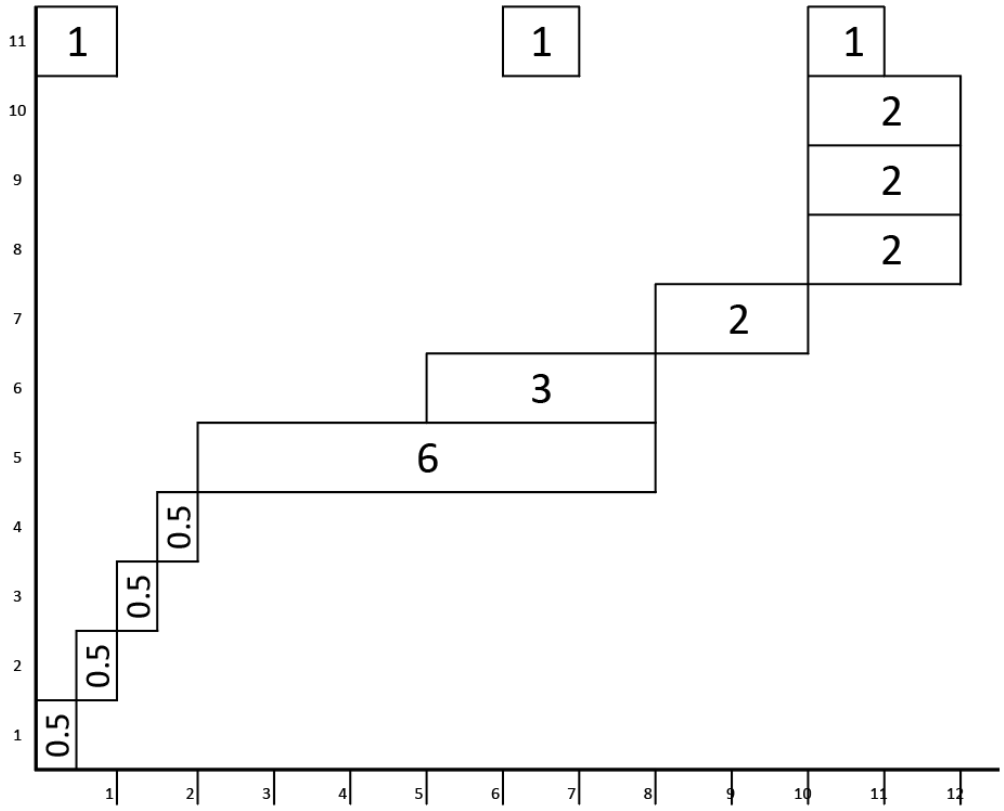


Рис 4.2 Графік розподілення часу

Баланс: 0,7 – 1,3

Таблиця 4.4 Загруженість по часу кожного з учасників

1 2 3 4	0.5 + 0.25 + 0.25 + 0.25 + 6 + 2 + 1 + 2 = 12.25 0.5 + 0.25 + 0.25 + 5 + 1 + 2 = 9 0.5 + 0.5 + 0.5 + 2 + 2 + 2 + 2 + 2 = 11.5 0,5 + 0.25 + 2 + 2 + 2 + 1 + 2 = 9.75	12,25 / 12 = 1.02 9 / 12 = 0.75 11.5 / 12 = 0.96 9.75 / 12 = 0,8125	1.375 / 1.02 = 1,345 1.5 / 0,75 = 2 0,77 / 0.96 = 0.8 0.875 / 0.8125 = 1.08

Таблиця 4.5 Визначення важливості факторів щодо їх вкладу у створення та реалізацію стартапу

Фактор	**Вага (важливість)**
Ідея	8
Підготовка бізнес плану	7
Компетентність	10
Залученість і ризики	6

Обов'язки	8
Залучення партнерів	5
Новизна ідеї	10

Таблиця 4.6 Оцінювання особистого внеску кожного партнера у створення та реалізацію стартапу

Фактор	**Вага**	**Партнер 1**	**Партнер 2**	**Партнер 3**	**Партнер 4**
Ідея	8	8	0	6	0
Підготовка бізнес плану	7	4	2	9	5
Компетентність	10	8	5	7	8
Залученість і ризики	6	5	5	6	6
Обов'язки	8	5	5	8	7
Залучення партнерів	5	0	0	5	8

Таблиця 4.7 Визначення дольової участі у стартап проекті кожного учасника

Фактор	**Партнер 1**	**Партнер 2**	**Партнер 3**	**Партнер4**	
Ідея	64	0	48	0	
Підготовка бізнес плану	28	14	63	35	
Компетентність	80	50	70	80	
Залученість і ризики	30	30	36	36	
Обов'язки	40	40	64	56	
Залучення партнерів	0	0	25	40	
Разом	242	134	306	247	929
Процент	26,04%	14,42%	32,93%	26.58%	100,0%

Таблиця 4.8 Морфологічна карта

Основні параметри	Проміжні рішення				
	1-ше	2-е	3-е	4-е	5-е
Реалізація інтерфейсу	Термінал, завантаження фото через шлях до файлу	Звичайна програма з GUI та вибором файлів	Зручний інтерфейс з багатьма місцями для перегляду зображень	Зручний веб-інтерфейс з можливістю перегляду з різних пристроїв	інше
Кількість одночасних камер	1	4	8	16	32+
Точність	50%	80%	90%	99%	146%

Темним затіненням показано оптимальні значення основних параметрів, більш світлим – допустимі значення.

Таким чином, ідею нового продукту можна сформулювати так: додаток стає швидким та зручним у використанні з додаванням нових особливостей для товару.

Задум товару:

1. Товар за задумом. Програма підтримує велику кількість можливостей для забезпечення безпеки користувача. Зручність використання додатку забезпечена підтримкою великої кількості камер, а також можливість виводити на екран імовірні місцезнаходження зброї.

2. Товар у реальному виконанні. Програма підтримує можливість виводити на окремий екран зображення, на якому ймовірно наявна зброя, що полегшує операторові роботу.

3. Товар з підкріпленням. Криптогаманець може бути налаштований та змінений під вимоги та потреби замовника. За бажанням можна змінювати кількість доступних камер, та додавати та налаштовувати інші камери.

4.3 Етапи еволюції захисту громадського порядку за методом MVP

MVP – це підхід що базується на передумові, що ви можете забезпечити достатню цінність для споживачів, надаючи мінімальні можливості, якими користуватимуться перші користувачі.[51]

Головна проблема: зручність та надійність

MVP1	MVP2	MVP3	MVP4	MVP5	MVP6
Двері	Замки	Пастки	Охоронці	Відео-камери	Алгоритми

Рис 4.2 MVP захисту даних

Головна проблема: зручність та надійність. Користувач має нативно орієнтуваися в додатку, а також передбачати всі моменти, де на кадрах з'являється зброя

1-й MVP: Двері. Двері та стіни використовувалися, як межа території, за яку не можна зайти.

2-й MVP: Замки. З'явилися замки, які могли обмежити доступ до того чи іншого предмету чи людини

3-й MVP: Пастки. Зросли ризики для зловмисника скоювати злочин. Міг поплатитися здоров'ям або життям.

4-й MVP: Охоронці. В процес включили людей, що відповідають за обмеження, але з цим зріс і людський фактор.

5-й MVP: Камери відеоспостереження. Зручність у використанні збільшилась, кількість працівників зменшилась за рахунок можливості перегляду одним спостерігачем. Людський фактор знову виріс.

6-й MVP: Автоматичні камери відеоспостереження. Система, що допомагає зменшити людський фактор. Значно збільшилася надійність та ефективність системи.

4.4 Ідеї та агрегування стартапу

Ідея 1. Камери що розпізнають холодну зброю.

Ідея 2. . Камери, що розпізнають вогнепальну зброю.

Ідея 3. Інфрачервоні камери для розпізнавання витоків з труб.

Ідея 4. Нейромережа, що буде на основі зображень з камери перевіряти труби на наявність щілин, і витоків.

Ідея 5. Програма тотального контролю, з датчиками на кожному носієві для забезпечення громадського порядку.

Ідея 6. Звуковий датчик щільності, що на основі коливань визначає зміну щільності матеріалу, що дає ймовірне місцезнаходження розриву.

Ідея 1. Камери що розпізнають холодну зброю.

- Дає змогу розпізнати зловмисника за рахунок наявності холодної зброї в руках
- Можливість розпізнавати мечі, мачете, ножі, кинджали, сокири.

Ідея 2. . Камери, що розпізнають вогнепальну зброю

- Дає змогу розпізнавати зловмисника до перших постраждалих
- Можливість розпізнавати пістолети, автомати, помпи.

Ідея 3. Інфрачервоні камери для розпізнавання витоків з труб.

- Дає змогу розпізнавати нештатні ситуації типу витоків води та газу, що протікають по трубах
- Також на деяких типах електростанцій мають можливість ідентифікувати деякі іншого роду витоки.

Ідея 4. Нейромережа, що буде на основі зображень з камери перевіряти труби на наявність щілин та витоків.

- Розпізнавання буде проходити у відеопотоці, де будуть порівнюватися кадри між собою, що дасть змогу розпізнати зміну об’єктів
- Також дає ефективність реакції і не потребує оператора, який мусив би постійно спостерігати

Ідея 5. Програма тотального контролю, з датчиками на кожному носієві для забезпечення громадського порядку.

- Підвищена кількість датчиків, що можуть спостерігати за зловмисниками
- Розподіляє навантаження обчислень на багато різних пристроїів.

Ідея 6. Звуковий датчик щільності, що на основі коливань визначає зміну щільності матеріалу, що дає ймовірне місцезнаходження розриву.

- Дає можливість спостерігати за критично важливими місцями в різного типу структурах
- Як недолік – може не надавати точність через зайві шуми від інших робітників компанії

Агрегування 1. Камери, що розпізнають розриви, що можуть бути в деяких місцях «візуально» розпізнавати різного типу витоки за допомогою зображення та інфрачервоних даних

Агрегування 2. Камери, що на основі нейронних мереж дані з камер відеоспостереження та інфрачервоних даних будуть досліджувати людину з середини на наявність зброї.

Агрегування 3. Звуковий датчик, який буде працювати тільки тоді, коли камера відеоспостереження та інфрачервоний датчик дадуть якісь позитивні результати, що дасть зменшення навантаження та збільшить точність

Надалі потрібно сформулювати та синхронізувати завдання.

Таблиця 4.9 Синхронізація завдань

Етапи	*Продукти (послідовність заміщення)*		
Минуле століття	Замки	Охоронці	
Сьогодні	Ідея 2. . Камери, що розпізнають вогнепальну зброю.	Ідея 1. Камери що розпізнають холодну зброю.	
Завтра	Ідея 3. Інфрачервоні камери для розпізнавання витоків з труб.	Ідея 4. Нейромережа, що буде на основі зображень з камери перевіряти труби на наявність щілин, і витоків.	Ідея 6. Звуковий датчик щільності, що на основі коливань визначає зміну щільності матеріалу, що дає ймовірне місцезнаходження розриву.
Післязавтра	Агрегування 3. Звуковий датчик, який буде працювати тільки тоді, коли камера відеоспостереження та інфрачервоний датчик дадуть якісь позитивні результати, що дасть зменшення навантаження та збільшить точність	Агрегування 1. Камери, що розпізнають розриви, що можуть бути в деяких місцях «візуально» розпізнавати різного типу витоки за допомогою зображення та інфрачервоних даних	Агрегування 2. Камери, що на основі нейронних мереж дані з камер відеоспостереження та інфрачервоних даних будуть досліджувати людину з середини на наявність зброї.
Відновлення інформації XXI століття	Ідея 5. Програма тотального контролю, з датчиками на кожному носієві для забезпечення громадського порядку.		

4.5 Розроблення ринкової стратегії проекту

Таблиця 4.10 Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Клієнти, що мають приватну власність, нерухомість	Середня, адже потрібен час для вироблення довіри до продукту	70%	Середня, не так багато додатків існує на ринку.	Середня, адже потрібен час для асиміляції з продуктом
2	Організації бізнес, та критичні інфраструктури	Висока, спеціалісти відразу побачать перспективу в безпеці керуванням транзакціями.	80%	Висока, за увагу бізнесу бориться велика кількість компаній.	Низька, адже є багато різних інших методів, які вже реалізовані і чудово працюють
Які цільові групи обрано: Фінансові інвестори – люди, що бажають швидко та безпечно керування електронними транзакціями.					

Таблиця 4.11 Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку*
	Переорієнтація проекту в аналіз зображень медичного	Стратегія концентрованого ринку (конкретний	Низька кількість аналогів, і напрямок на найдорожче, що є у людей	Стратегія удосконалення

	напрямку, щоб виявляти різні критичні ситуації в житті людей	сегмент споживачів)		

Таблиця 4.12 Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
	Ні	Так, так	Так, а саме можливість ідентифікації зображень на різних об’єктах.	Стратегія заняття конкурентної ніші

Таблиця 4.13 Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
	Зручність, UI/UX Швидкість розпізнавання Точність розпізнавання Надійність	Стратегія удосконалення	- Зручність у використанні - Немає прямих аналогів, що включають всі аспекти продукту - Продукт має змогу перерости в культовий	Швидкість Надійність

4.6 Розроблення маркетингової програми стартап-проекту

Таблиця 4.14 Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
	Забезпечити безпеку на об’єктах критичної інфраструктури	Підтримка декількох камер відеоспостереження. Швидкість роботи нейронної мережі. Забезпечення захисту від програмного вторгнення.	Можливість спостереження за різними типами нештатних ситуацій. Швидкість роботи системи. Робота з різного типами датчиків.

Таблиця 4.14 Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Програма надає можливості купівлі криптовалют, а також операціями над та між існуючими активами.		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Можливість надійно зберігати кошти.	Нм	Тл
	2. Підтримка не тільки найбільш популярної криптовалюти Bitcoin. Але й багатьох інших.	Нм	Тх
	3.Не тільки зберігати всі акти в одному місці, але й мати змогу виконувати обмін між наявними криптовалютами.	Нм	Тх
	4. Естетичні – привабливий дизайн	Нм	Е
	Якість: UI/ UX стандарти, високий code coverage, ефективне шифрування данних споживачів, швидкодія транзакцій та системи в цілому, висока якість технічної підтримки		

	Пакування: відсутнє – це та онлайн додаток
	Марка: Coinplace
III. Товар із підкріпленням	Криптогаманець може бути налаштований та змінений під вимоги та потреби замовника. За бажанням можна регулювати ключові параметри системи (кількість криптовалют, кількість потрібних приватних ключів, що в свою чергу змінює вимоги до програмного забезпечення).
За рахунок чого потенційний товар буде захищено від копіювання: За рахунок прав на інтелектуальний ресурс.	

Таблиця 4.14 Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
	5000 грн в рік	8000 грн	10 000 грн	Додаток безкоштовний, комісія за знаходження однієї одиниці зброї 100 грн

Таблиця 4.15 Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
	Клієнти виконують транзакції через додаток веб-сайт.	Канал збуту – мережа інтернет.	0	Власний landing page

Концепція маркетингових комунікацій

В моєму випадку це товар, що виступає на пряму контакт з клієнтами та компаніями, тому комунікації з потенційними клієнтами на мій погляд можливі тільки особисті (на бізнес форумах, технічних форумах).

4.7 Аналіз ринкових можливостей запуску стартап-проекту

Таблиця 4.15 Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	6
2	Загальний обсяг продаж, грн/ум.од	2 625 000 грн
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Довіра клієнтів
5	Специфічні вимоги до стандартизації та сертифікації	немає
6	Середня норма рентабельності в галузі (або по ринку), %	105 * 25 / 28000 = 21.85 %

Таблиця 4.16 Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
	Швидкість та надійність визначення.	Окремі організації та користувачі, що бажають надійно зберігати своїх працівників та приватну власність або нерухомість.	Кількість об’єктів. Мається на увазі точки, в яких є необхідність встановлення камер.	- Наявність зору для того, щоб бачити.

Таблиця 4.17 Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Мала кількість спроможних користувачів	Існує можливість, що користувачі не зможуть купувати продукт.	Стимулювання довіри, рекламна кампанія.

2.	Поява нових конкурентів	Зниження кількості клієнтів	Створення нових можливостей, що буде конкурувати з ними.
3.	Криза в країні	Зниження платоспроможності клієнтів	Перехід на міжнародну арену.

Таблиця 1.18 Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Залучення великої кількості користувачів	Якість продукту даватиме рекламу особисто від особи до особи.	Стимулювання користувачів ще більше довіряти продукту, система лояльності.
2	Вихід на міжнародну арену	Залучення міжнародних інвесторів, та зростання популярності на міжнародній арені.	Відкриття нових міжнародних офісів. Стимулювання нових клієнтів.

Таблиця 4.19 Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції - монополія/олігополія/ монополістична/чиста	олігополія	Залучення сучасних технологій, взаємодія з користувачами, опитування щодо продукту. Прийняття участі у бізнес-конференціях.
2. За рівнем конкурентної боротьби - локальний/національний/ …	Національно-міжнародний.	Розвиток на національному ринку, а також вихід на міжнародну арену.

3. За галузевою ознакою - міжгалузева/ внутрішньогалузева	Внутрішньогалузева	Конкуренція відбується між існуючими компаніями.
4. Конкуренція за видами товарів: - товарно-родова - товарно-видова - між бажаннями	Товарно-видова	Через наявність аналогів буде зберігатися олігополія
5. За характером конкурентних переваг - цінова / нецінова	цінова	Підписка на програмний продукт
6. За інтенсивністю - марочна/не марочна	Марочна	Вкладатись в впізнаваність бренду

Таблиця 4.20 Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
Складові аналізу	Сіріус	Немає	Постачальники надають в оренду сервери	Споживачі мають безпеку в своїх домівках та бізнесі	Служби безпеки
Висновки:	Ні, адже конкуренти спеціалізуються переважно на сиганлізаціях, які незалежні від камер.	- є можливість виходу на ринок	Постачальники не диктують умов роботи на ринку	Так, тільки за допомогою відгуків.	Можлива недовіра до камер, адже більшість людей звикли до сигналізацій

Таблиця 4.21 Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)

	Існування підтримки багатьох камер відеоспостереження, які можуть виявляти різного типу нештатні ситуації.	Набору таких можливостей у конкурентів не існує.

Таблиця 4.22 Порівняльний аналіз сильних та слабких сторін

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з … (назва підприємства)						
			–3	–2	–1	0	+1	+2	+3
1	Надійність	19		х	х				
2	Швидкість	18						Х	
3	Зручність	18					х		

Таблиця 4.23 SWOT- аналіз стартап-проекту

Сильні сторони: орієнтація на фінансово-спроможних людей,	Слабкі сторони: новий сервіс, що може користуватися не довірою у клієнтів.
Можливості: сервіс буде зручнішим і замовники віддадуть йому перевагу. Залучення міжнародних бізнесменів.	Загрози: Недовіра від клієнтів, зменшення кількості правопорушень, врегулювання законів, що сприятимуть цьому. Рішення проблеми на державному рівні.

Таблиця 4.24 Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
	Відмова від самостійного сервісу на користь інтеграції програмного продукту в існуючі системи оплати	Висока	5 місяців.

Таблиця 4.25 Профіль посади. Партнер по бізнесс - частині.

Критерій	Зміст
Основна освіта	Магістр
Додаткова освіта, спеціалізація	Освітні програми по розвитку бізнесу.
Необхідний досвід роботи	Досвід успішних проектів
Завдання	Знання сучасного ринку
Знання	Просування інноваційних продуктів
Навички, вміння, ділові якості	Відповідальність.
Особистісні якості	
Мотивація (що можемо запропонувати)	Частина в стартапі.

Нематеріальне стимулювання: для технічних співробітників – можливість працювати з сучасними технологіями, цікава ідея продукту, можливість працювати віддалено.

Для всіх співробітників нематеріальне стимулювання – зручний офіс в центрі, медичне страхування, сімейна атмосфера, тімбілдинги.

Матеріальне стимулювання – заробітна плата + річний бонус при досягненні необхідних kpi. Для менеджерів з продажу є мінімальна місячна ставка 10000 грн та відсоток від продажів. Для ключових фігур – можливість придбати акції компанії.

Висновки до розділу 4

Даний розділ був присвячений розробці стартап проекту для системи ідентифікації нештатних ситуацій на об’єктах критичної інфраструктури за допомогою нейронних мереж. Для розробки стартап проекту було виконано наступні дії:

- Було описано інформаційну карту проекту, де була вказана вся необхідна інформація щодо дипломного проекту.
- Описаний принцип та план роботи з командою стартапу, що складається з 4-х партнерів: IT-спеціаліст генератор ідей, IT-спеціаліст виконавець, менеджер-координатор та маркетолог що виконує функції дипломату.
- Було описано етапи еволюції захисту громадського порядку за методом MVP. Від дверей та замків до сучасних алгоритмів ідентифікації нештатних ситуацій.
- Описані ідеї реалізації продукту, а також втілення їх в агрегуванні декількох з них.
- Була розроблена ринкова стратегія проекту.
- А також була створена маркетингова програма стартап-проекту.

В результаті, було отримано стартап проект для запуску розробленого програмного забезпечення на ринок, отримано навички створення стартап-проектів, роботи з командою, побудови стратегії маркетингу і аналізу обраного ринку.

ВИСНОВКИ

Було розглянуто роботу власного продукту а також порівняння з деякими існуючими. Можна зробити висновок, що існують і на багато ліпші варіанти систем, що можуть розпізнавати ручну зброю, таку як пістолети, гвинтівки, помпові рушниці, ножі, кинджали, і подібне. Але в основному системи, що мають більшу точність мають нижчу швидкість роботи. Тому використання системи архітектури нейронної мережі власної розробки в даній роботі являється максимально оптимальним. Також на відміну від інших версій алгоритму YOLO, за рахунок того, що власна розробка розбирає зображення на більшу кількість клітинок з більшою кількістю дрібних об'єктів, і при тому не шукає великі об'єкти, дана система працює оптимально добре для того, щоб використовувати її в системах відеоспостереження за злочинністю на об'єктах критичної інфраструктури.

Найбільшим недоліком системи являється те, що вона повільно працює відносно інших систем, хоча цей недолік чудово вирішується вирізанням певних кадрів із відео потоку, що надається на обробку нейронній мережі. Тобто при роботі системи можна надсилати безпосередньо ту кількість кадрів, яку вона має змогу встигнути обробити за той час.

У порівнянні з існуючими версіями алгоритму YOLO, власна розробка може працює швидше ніж YOLO третьої версії, зберігаючи його ефективність у власній задачі. Таким чином власна розробка працює швидше ніж YOLO третьої версії, з не значним зниженням ефективності, а також ефективніше ніж YOLO другої версії, але з не значним зниженням швидкості.

Магістерська дипломна робота надала можливість детально вивчити технології комп'ютерного зору, нейронних мереж, а в особливості ідентифікацію об'єктів за рахунок нейронних мереж архітектури YOLO другої та третьої версії.

В майбутньому планується розробити ідентифікацію нештатних ситуацій наприклад проривів труб на заводах або гідроелектростанціях за рахунок

комп’ютерного зору. Таким чином камери відеоспостереження та дану розробку можна буде використовувати з різними цілями.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Do you know how the X-ray device at airport security screening works? [Електронний ресурс] – Режим доступу: https://www.finavia.fi/en/newsroom/2018/do-you-know-how-x-ray-device-airport-security-screening-works#:~:text=The%20conveyor%20belt%20carries%20each,progression%E2%80%9D%2C%20explains%20Mikko%20Halonen
2. Radiation and Airport Security Scanning [Електронний ресурс] – Режим доступу: https://www.epa.gov/radtown/radiation-and-airport-security-scanning
3. Wade Deisman. CCTV: Literature Review and Bibliography. – 2003.
4. Michał Grega, Andrzej Matiolański, Piotr Guzik, and Mikołaj Leszczuk. Automated detection of firearms and knives in a cctv image. – 2016 – С. 1–2.
5. Kristi Targamaa Mareks Smilga Kgstutis Grahlis, Dainius Barkauskas Pavel Laptev. MINIMUM REQUIREMENTS FOR VIDEO SURVEIL-LANCE SYSTEM FOR PUBLIC AREAS. – 2016 – С. 1–2.
6. Balancing Utilities: Public Video Surveillance in Sweden. – 2010 – С. 4–5, 8–10, 12
7. How many CCTV Cameras are there in London? [Електронний ресурс] Режим доступу: https://www.cctv.co.uk/how-many-cctv-cameras-are-there-in-london/
8. Aleksandra Maksimova. Knife detection scheme based on possibilistic shell clustering. – 2013 – С. 1
9. Andrzej Glowacz, Marcin Kmieć, and Andrzej Dziech. Visual detection of knives in security applications using active appearance models. – 2015 – С. 4–5
10. C. Anagnostopoulos, I. Anagnostopoulos, G. Tsekouras, G. Kouzas, V. Loumos, and E. Kayafas. Using sliding concentric windows for license plate segmentation and processing. – 2005. – С. 337–342
11. Nash W, Drummond T, Birbilis N A Review of Deep Learning in the Study of Materials Degradation. – 2018 – [Електронний ресурс]. Режим доступу: https://doi.org/10.1038/s41529-018-0058-x

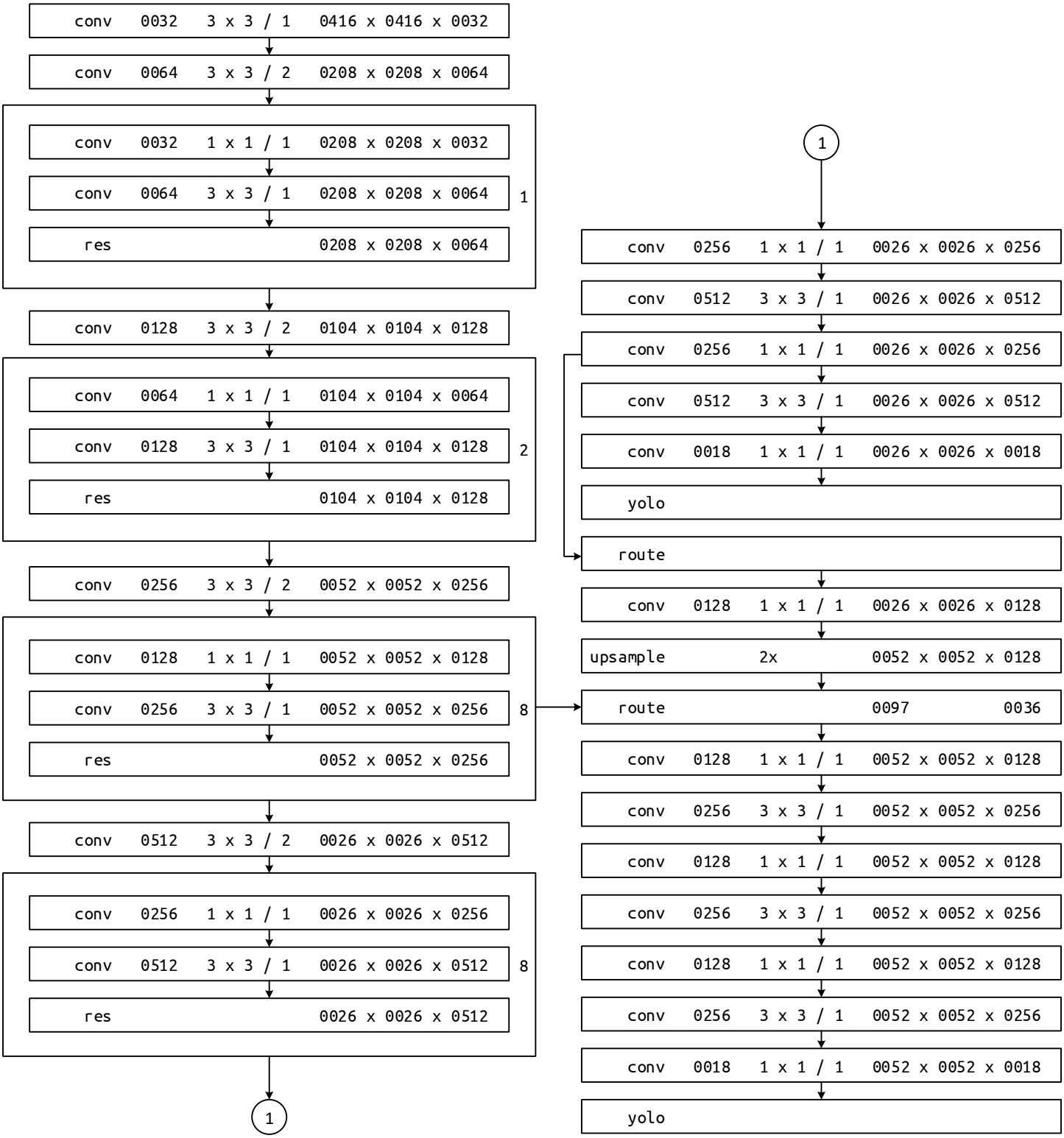
12. Bonaccorso G. Machine Learning Algorithms Popular Algorithms for Data Science and Machine Learning. – 2018
13. O'Mahony N, Murphy T, Panduru K, et al. Real-time monitoring of powder blend composition using near infrared spectroscopy. – 2017 – C. 1–6
14. O' Mahony N, Murphy T, Panduru K. Adaptive process control and sensor fusion for process analytical technology. – 2016 – C. 1–6
15. Koehn P, Koehn P Combining Genetic Algorithms and Neural Networks: The Encoding Problem – 2011 – C. 2
16. Niall O' Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, Joseph Walsh. Deep Learning vs. Traditional Computer Vision. – 2018 – C. 3–4
17. Voulodimos A, Doulamis N, Doulamis A, Protopapadakis E. Deep Learning for Computer Vision: A Brief Review. Comput Intell Neurosci. – 2018 – C.1–13. [Електронний ресурс], Режим доступу: https://doi.org/10.1155/2018/7068349
18. Dumoulin V, Visin F, Box GEP. A guide to convolution arithmetic for deep learning. – 2018 – C.1–3
19. Horiguchi S, Ikami D, Aizawa K. Significance of Softmax-based Features in Comparison to Distance Metric Learning-based Features. – 2018 – C. 1–2
20. Wang J, Perez L The Effectiveness of Data Augmentation in Image Classification using Deep Learning. – 2018 – C. 1–3
21. Schöning J, Faion P, Heidemann G. Pixel-wise Ground Truth Annotation in Videos - An Semi-automatic Approach for Pixel-wise and Semantic Object Annotation. – 2016 – C. 2
22. Zhang X, Lee J-Y, Sunkavalli K, Wang Z Photometric Stabilization for Fastforward Videos – 2017 – C.2
23. Alhaija HA, Mustikovela SK, Mescheder L. Augmented Reality Meets Computer Vision : Efficient Data Generation for Urban Driving Scenes. – 2017 – C.2
24. Meneghetti G, Danelljan M, Felsberg M, Nordberg K. Image Alignment for Panorama Stitching in Sparsely Structured Environments. – 2015 – C.3

25. Alldieck T, Kassubeck M, Magnor M. Optical Flow-based 3D Human Motion Estimation from Monocular Video – 2017 – C.2
26. Zheng B, Zhao Y, Yu J. Scene Understanding by Reasoning Stability and Safety. – 2015 – C.2 – [Електронний ресурс], Режим доступу: https://doi.org/10.1007/s11263-014-0795-4
27. Karami E, Shehata M, Smith A. Image Identification Using SIFT Algorithm: Performance Analysis against Different Image Deformations. – 2017 – C.3
28. Bay H, Tuytelaars T, Van Gool L. SURF: Speeded Up Robust Features. Springer, Berlin, Heidelberg. – 2006 – C.1–2
29. Rosten E, Drummond T. Machine Learning for High-Speed Corner Detection. Springer, Berlin, Heidelberg. – 2006
30. Goldenshluger A, Zeevi A. The Hough Transform Estimator. – 2004 – C.4 – [Електронний ресурс] Режим доступу: https://doi.org/10.1214/009053604000000760
31. Tsai FCD. Geometric hashing with line features. Pattern Recognit. – 1994 – C.377–389. [Електронний ресурс] Режим доступу: https://doi.org/10.1016/0031-3203(94)90115-5
32. European Programme for Critical Infrastructure Protection. – 2007 – [Електронний ресурс] Режим доступу: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=LEGISSUM:l33260
33. Alan T. Murray, Tony H. Grubesic. Overview of Reliability and Vulnerability in Critical Infrastructure. – 2007 – C.93
34. Munaisyah Abdullah, Muhammad Yahya, Mohd Nizam Husen. Automatic Handgun and Knife Detection Algorithms. – 2020 – C. 1–7
35. D. KG. The Harris corner detector. – 2004. – C. 1–7
36. A. Glowacz, M. Kmieć and A. Dziech, Visual detection of knives in security applications using Active Appearance Models. – 2013 – C. 12

37. M. Nakib, T. K. Rozin and M. S. Hasan. Crime Scene Prediction by Detecting Threatening Objects Using Convolutional Neural Network, – 2017. – C. 2–3, 8–10, 12
38. P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. – 2013. – C. 8–9
39. Joseph Redmon. You Only Look Once: Unified, Real-Time Object Detection. – 2016 – C. 15–23.
40. Eli Stevens, Luca Antiga, Thomas Viehmann. Deep Learning with PyTorch. – 2020 – C. 7–8
41. JV Dillon, I Langmore, D Tran, E Brevdo. Tensorflow distributions. – 2018 – C. 1–2
42. Jupyter architecture [Електронний ресурс]. Режим доступу: https:// jupyter.readthedocs.io/en/latest/projects/architecture/content-architecture.html
43. T. Rohit Kumar and V. Gyanendra K. A Computer Vision based Framework for Visual Gun Detection using Harris Interest Point Detector. – 2015 – C.711
44. A. Glowacz, M. Kmieć and A. Dziech. Visual detection of knives in security applications using Active Appearance Models. – Multimedia Tools and Applications, – 2013 – C.4261.
45. P. Pratihar and A. K. Yadav. Detection Techniques for Human Safety from Concealed weapon and Harmful EDS. – International Review of Applied Engineering Research. – 2017 – C.71–76.
46. Roberto Olmos. Automatic Handgun Detection Alarm in Videos Using Deep Learning. – 2017 – C.10, 14.
47. L. Justin and M. Sydney. Developing a Real-Time Gun Detection Classifier. – 2018 – C.3–4.
48. G. Verma and A. Dhillon. A Handheld Gun Detection using Faster R-CNN Deep Learning. – 2017 – C.4–5.

49. M. Nakib, T. K. Rozin and M. S. Hasan, Crime Scene Prediction by Detecting Threatening Objects Using Convolutional Neural Network. – 2017 – С.26–31.
50. Z. Jinsong, X. Wenjie, X. Mengdao and S. Guangcai, "Terahertz Image Detection with the Improved Faster. – 2018 – С.15-16.
51. Product development: Minimum viable product (MVP) approach [Електронний ресурс]. Режим доступу: https://learn.marsdd.com/article/product-development-minimum-viable-product-mvp-approach/

ДОДАТОК 1. АРХІТЕКТУРА НЕЙРОННОЇ МЕРЕЖІ



ДОДАТОК 2. ЛІСТИНГ ПРОГРАМИ

In [1]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=94731
8989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3
aietf%3awg%3aoauth%3a2.0%3aoob&response_type=code&scope=email%20https%3a%2f%2fwww.goog
leapis.com%2fauth%2fdocs.test%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive%20http
s%3a%2f%2fwww.googleapis.com%2fauth%2fdrive.photos.readonly%20https%3a%2f%2fwww.google
apis.com%2fauth%2fpeopleapi.readonly

Enter your authorization code:
··········
Mounted at /content/drive
```

In [2]:

```
!nvcc --version
```

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2019 NVIDIA Corporation
Built on Sun_Jul_28_19:07:16_PDT_2019
Cuda compilation tools, release 10.1, V10.1.243
```

In [3]:

```
!unzip "/content/drive/My Drive/YOLOv3_files/darknet.zip"
```

```
Streaming output truncated to the last 5000 lines.
inflating: darknet/data/images/12_00025_00011.ppm.jpg
inflating: darknet/data/images/12_00025_00012.ppm.jpg
inflating: darknet/data/images/12_00025_00013.ppm.jpg
inflating: darknet/data/images/12_00025_00014.ppm.jpg
inflating: darknet/data/images/12_00025_00015.ppm.jpg
…
inflating: darknet/data/labels/14_00025_00027.ppm.txt
inflating: darknet/data/labels/14_00025_00028.ppm.txt
inflating: darknet/data/labels/14_00025_00029.ppm.txt
```

In [4]:

```
%cd /content/darknet
!make
!chmod +x ./darknet
```

```
/content/darknet
mkdir -p obj
mkdir -p backup
mkdir -p results
gcc -Iinclude/ -Isrc/ -DOPENCV `pkg-config --cflags opencv`  -DGPU -I/usr/local/cuda/i
nclude/ -DCUDNN  -Wall -Wno-unused-result -Wno-unknown-pragmas -Wfatal-errors -fPIC -O
fast -DOPENCV -DGPU -DCUDNN -c ./src/gemm.c -o obj/gemm.o
./src/gemm.c: In function ‘time_gpu’:
./src/gemm.c:232:9: warning: ‘cudaThreadSynchronize’ is deprecated [-Wdeprecated-decla
rations]
```

```
          cudaThreadSynchronize();
          ^~~~~~~~~~~~~~~~~~~~
In file included from /usr/local/cuda/include/cuda_runtime.h:96:0,
                 from include/darknet.h:11,
                 from ./src/utils.h:5,
                 from ./src/gemm.c:2:
/usr/local/cuda/include/cuda_runtime_api.h:957:57: note: declared here
 extern __CUDA_DEPRECATED __host__ cudaError_t CUDARTAPI cudaThreadSynchronize(void);
                                                         ^~~~~~~~~~~~~~~~~~~~
gcc -Iinclude/ -Isrc/ -DOPENCV `pkg-config --cflags opencv`  -DGPU -I/usr/local/cuda/i
nclude/ -DCUDNN  -Wall -Wno-unused-result -Wno-unknown-pragmas -Wfatal-errors -fPIC -O
fast -DOPENCV -DGPU -DCUDNN -c ./src/utils.c -o obj/utils.o
…
```

In [0]:

```
!rm /content/darknet/backup -r
!ln -s /content/drive/'My Drive'/YOLOv3_weight/backup /content/darknet
```

In [6]:

```
!sudo apt install dos2unix
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  dos2unix
0 upgraded, 1 newly installed, 0 to remove and 31 not upgraded.
Need to get 351 kB of archives.
After this operation, 1,267 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/universe amd64 dos2unix amd64 7.3.4-3 [3
51 kB]
Fetched 351 kB in 0s (4,007 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend can
not be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 76, <> line 1.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package dos2unix.
(Reading database ... 144439 files and directories currently installed.)
Preparing to unpack .../dos2unix_7.3.4-3_amd64.deb ...
Unpacking dos2unix (7.3.4-3) ...
Setting up dos2unix (7.3.4-3) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

In [7]:

```
!dos2unix ./data/train.txt
!dos2unix ./data/val.txt
!dos2unix ./data/yolo.data
!dos2unix ./data/yolo.names
!dos2unix ./cfg/yolov3_custom_train.cfg
dos2unix: converting file ./data/train.txt to Unix format...
dos2unix: converting file ./data/val.txt to Unix format...
```

```
dos2unix: converting file ./data/yolo.data to Unix format...
dos2unix: converting file ./data/yolo.names to Unix format...
dos2unix: converting file ./cfg/yolov3_custom_train.cfg to Unix format...
```

In [9]:

```
# If your output cell shows something similar like belows, then congratulation, your
model has started training successfully!
%cd /content/darknet
!./darknet detector train data/yolo.data cfg/yolov3_custom_train.cfg darknet53.conv.7
4
```

…

In [0]:

```
# copy pre-trained weight from your Drive to folder darkner
!cp /content/drive/'My Drive'/YOLOv3_files/yolov3_900.weights /content/darknet

# train annd save loss result in log.txt
%cd /content/darknet
!./darknet detector train data/yolo.data cfg/yolov3_custom_train.cfg yolov3_900.weigh
ts | tee log.txt
```